

Documented Code For glossaries v4.0

Nicola L.C. Talbot

Dickimaw Books

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

2013-11-14

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 v4.0: L^AT_EX2e Package to Assist Generating Glossaries”.

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

[glossaries-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

1 Main Package Code	3
1.1 Package Definition	3
1.2 Package Options	5
1.3 Default values	25
1.4 Xindy	34
1.5 Loops and conditionals	43
1.6 Defining new glossaries	46
1.7 Defining new entries	49
1.8 Resetting and unsetting entry flags	69
1.9 Loading files containing glossary entries	71
1.10 Using glossary entries in the text	72
1.10.1 Links to glossary entries	79
1.10.2 Displaying entry details without adding information to the glossary	131
1.11 Adding an entry to the glossary without generating text	137
1.12 Creating associated files	138
1.13 Writing information to associated files	148
1.14 Glossary Entry Cross-References	152
1.15 Displaying the glossary	154
1.16 Acronyms	171
1.17 Predefined acronym styles	175
1.18 Predefined Glossary Styles	194
1.19 Debugging Commands	195
1.20 Compatibility with version 2.07 and below	199
2 Prefix Support (glossaries-prefix Code)	200
3 Mfirstuc Documented Code	206
4 Glossary Styles	209
4.1 Glossary hyper-navigation definitions (glossary-hypernav package)	209
4.2 In-line Style (glossary-inline.sty)	211
4.3 List Style (glossary-list.sty)	213
4.4 Glossary Styles using longtable (the glossary-long package)	216
4.5 Glossary Styles using longtable (the glossary-longragged package)	223
4.6 Glossary Styles using multicol (glossary-mcols.sty)	228
4.7 Glossary Styles using supertabular environment (glossary-super package)	232
4.8 Glossary Styles using supertabular environment (glossary-superragged package)	239
4.9 Tree Styles (glossary-tree.sty)	244
5 glossaries-compatible-207	252

6 Accessibility Support (glossaries-accsupp Code)	272
6.1 Defining Replacement Text	273
6.2 Accessing Replacement Text	276
6.3 Displaying the Glossary	287
6.4 Acronyms	289
6.5 Debugging Commands	292
7 Multi-Lingual Support	293
7.1 Babel Captions	294
7.2 Polyglossia Captions	299
7.3 Brazilian Dictionary	303
7.4 Danish Dictionary	303
7.5 Dutch Dictionary	303
7.6 English Dictionary	304
7.7 French Dictionary	304
7.8 German Dictionary	304
7.9 Irish Dictionary	304
7.10 Italian Dictionary	305
7.11 Magyar Dictionary	305
7.12 Polish Dictionary	305
7.13 Serbian Dictionary	306
7.14 Spanish Dictionary	306
Glossary	306
Change History	306
Index	321

1 Main Package Code

1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

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

The textcase package has much better case changing handling, so use `\MakeTextUppercase` instead of `\MakeUppercase`

```
6 \RequirePackage{textcase}
7 \renewcommand*{\mfirstucMakeUppercase}{\MakeTextUppercase}%
```

```
8 \RequirePackage{xfor}
```

```
9 \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.)

```
10 \RequirePackage{amsgen}
```

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

```
11 \RequirePackage{etoolbox}
```

Check if doc has been loaded.

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

```
12 \newif\if@gls@docloaded
```

```
13 \@ifpackageloaded{doc}{%
```

```
14 {%
```

```
15   \@gls@docloadedtrue
```

```
16 }%
```

```
17 {%
```

```
18   \@ifclassloaded{nlctdoc}{\@gls@docloadedtrue}{\@gls@docloadedfalse}%
```

```
19 }
```

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

`\doc` 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
```

```
21 \newcommand{\glsorg@glossary}{%
```

```
22   \@bsphack
```

```
23   \begingroup
```

```
24     \@sanitize \endgroup\@esphack
```

```
25 }
```

```
\glsorg@wrglossary
```

```
26 \newcommand{\glsorg@wrglossary}[1]{%
```

```
27   \protected@write\@glossaryfile{}{%
```

```
28     \string \glossaryentry{#1}{\thepage}}%
```

```
29   \endgroup
```

```
30   \@esphack
```

```
31 }
```

```
32 \renewcommand*{\RecordChanges}{%
```

```
33   \newwrite\@glossaryfile
```

```
34   \immediate\openout\@glossaryfile=\jobname.glo
```

```
35   \def\glsorg@glossary{\@bsphack\begingroup\@sanitize\glsorg@wrglossary}%
```

```
36   \typeout{Writing glossary file \jobname .glo}%
```

```
37 }
```

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

```
38 \let\glsorg@changes\changes
39 \renewcommand{\changes}[3]{%
40   \begingroup
41     \let\glossary\glsorg@glossary
42     \glsorg@changes{#1}{#2}{#3}%
43   \endgroup
44 }
```

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

`\glsorg@theglossary`

```
45 \let\glsorg@theglossary\theglossary
```

`\glsorg@endtheglossary`

```
46 \let\glsorg@endtheglossary\endtheglossary
```

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

```
47 \let\glsorg@PrintChanges\PrintChanges
48 \renewcommand{\PrintChanges}{%
49   \begingroup
50     \let\theglossary\glsorg@theglossary
51     \let\endtheglossary\glsorg@endtheglossary
52     \glsorg@PrintChanges
53   \endgroup
54 }
```

End of doc stuff.

```
55 \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.

```
56 \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`.

```
57 \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`.

```
58 \ifcsundef{chapter}%
59   {\newcommand*\@@glossarysec{section}}%
60   {\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 redefine `\glossarysection`.

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

Determine whether or not to use numbered sections.

`\@@glossarysecstar`

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

`\@@glossaryseclabel`

```
65 \newcommand*{\@@glossaryseclabel}{}%
```

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

```
66 \newcommand*{\glsautoprefix}{}%
```

`numberedsection`

```
67 \define@choicekey{glossaries.sty}{numberedsection}[\val\nr]{%
68 false,nolabel,autolabel}[nolabel]{%
69   \ifcase\nr\relax
70     \renewcommand*{\@@glossarysecstar}{*}%
71     \renewcommand*{\@@glossaryseclabel}{}%
72   \or
73     \renewcommand*{\@@glossarysecstar}{}%
74     \renewcommand*{\@@glossaryseclabel}{}%
75   \or
76     \renewcommand*{\@@glossarysecstar}{}%
77     \renewcommand*{\@@glossaryseclabel}{}%
78     \label{\glsautoprefix\glo@type}%
79   \fi
80 }
```

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

`\@glossary@default@style`

```
81 \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](#).

```

82 \define@key{glossaries.sty}{style}{%
83   \renewcommand*{\@glossary@default@style}{#1}%
84 }

```

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

`\glossaryentrynumbers`

```

85 \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).

```

86 \DeclareOptionX{nonumberlist}{%
87   \renewcommand*{\glossaryentrynumbers}[1]{\gls@save@numberlist{#1}}%
88 }

```

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

```

89 \define@boolkey{glossaries.sty}[gls]{savenumberlist}[true]{}
90 \glssavenumberlistfalse

```

`\@seeautonumberlist`

```

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

```

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

```

92 \DeclareOptionX{seeautonumberlist}{%
93   \renewcommand*{\@glo@seeautonumberlist}{%
94     \def\@glo@prefix{\glsnextpages}%
95   }%
96 }

```

`\@gls@loadlong`

```

97 \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.

```

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

```

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

```

99 \IfFileExists{supertabular.sty}{%
100   \newcommand*{\@gls@loadsuper}{\RequirePackage{glossary-super}}{%
101   \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.

```

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

```

\@gls@loadlist

```

103 \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.

```

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

```

\@gls@loadtree

```

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

```

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

```

106 \DeclareOptionX{notree}{\renewcommand*{\@gls@loadtree}{}}

```

nostyles Provide an option to suppress all the predefined styles (in the event that the user has custom styles that are not dependent on the predefined styles).

```

107 \DeclareOptionX{nostyles}{%
108   \renewcommand*{\@gls@loadlong}{}%
109   \renewcommand*{\@gls@loadsuper}{}%
110   \renewcommand*{\@gls@loadlist}{}%
111   \renewcommand*{\@gls@loadtree}{}%
112   \let\@glossary@default@style\relax
113 }

```

\glspostdescription The description terminator is given by `\glspostdescription` (except for the 3 and 4 column styles). This is a full stop by default. The spacefactor is adjusted in case the description ends with an upper case letter. (Patch provided by Michael Pock.)

```

114 \newcommand*{\glspostdescription}{%
115   \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi
116 }

```

nopostdot Boolean option to suppress post description dot

```

117 \define@boolkey{glossaries.sty}[gls]{nopostdot}[true]{}
118 \glsnopostdotfalse

```

nogroupskip Boolean option to suppress vertical space between groups in the pre-defined styles.

```

119 \define@boolkey{glossaries.sty}[gls]{nogroupskip}[true]{}
120 \glsnogroupskipfalse

```


ucmark Boolean option to determine whether or not to use upper case in definition of `\glsglossarymark`

```
121 \define@boolkey{glossaries.sty}[gls]{ucmark}[true]{}
122 \@ifclassloaded{memoir}
123 {%
124   \glsucmarktrue
125 }%
126 {%
127   \glsucmarkfalse
128 }
```

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`.

```
129 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
130 \glsentrycounterfalse
```

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

```
131 \define@key{glossaries.sty}{counterwithin}{%
132   \renewcommand*{\@gls@counterwithin}{#1}%
133   \glsentrycountertrue
134 }
```

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

```
135 \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`.

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

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

```
138 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{%
139   \csname @gls@setupsort@#1\endcsname
140 }
```

`\glsprestandardsort` `\glsprestandardsort{<sort cs>}{<type>}{<label>}`

Allow user to hook into sort mechanism. The first argument `<sort cs>` is the temporary control sequence containing the sort value before it has been sanitized and had `makeindex/xindy` special characters escaped.

```
141 \newcommand*{\glsprestandardsort}[3]{%
142   \glsdosanitizesort
143 }
```

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

```

144 \newcommand*{\@gls@setupsort@standard}{%
    Store entry information when it's defined.
145   \def\do@glo@storeentry{\@glo@storeentry}%
    No count register required for standard sort.
146   \def\@gls@defsortcount##1{%
    Sort according to sort key (\@glo@sort) if provided otherwise sort according
    to the entry's name (\@glo@name). (First argument glossary type, second argu-
    ment entry label.)
147   \def\@gls@defsort##1##2{%
148     \ifx\@glo@sort\@glsdefaultsort
149       \let\@glo@sort\@glo@name
150     \fi
151     \let\glsdosanitizesort\@gls@sanitizesort
152     \glsprestandardsort{\@glo@sort}{##1}{##2}%
153     \expandafter\protected@xdef\csname glo@##2@sort\endcsname{\@glo@sort}%
154   }%
    Don't need to do anything when the entry is used.
155   \def\@gls@setsort##1{%
156 }
    Set standard sort as the default:
157 \@gls@setupsort@standard

```

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

```

158 \newcommand*\glssortnumberfmt[1]{%
159   \ifnum#1<100000 0\fi
160   \ifnum#1<10000 0\fi
161   \ifnum#1<1000 0\fi
162   \ifnum#1<100 0\fi
163   \ifnum#1<10 0\fi
164   \number#1%
165 }

```

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

```

166 \newcommand*{\@gls@setupsort@def}{%
    Store entry information when it's defined.
167   \def\do@glo@storeentry{\@glo@storeentry}%
    Defined count register associated with the glossary.
168   \def\@gls@defsortcount##1{%
169     \expandafter\global
170     \expandafter\newcount\csname glossary@##1@sortcount\endcsname
171   }%

```

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

```
172 \def\@gls@defsort##1##2{%
173   \expandafter\global\expandafter
174   \advance\csname glossary@##1@sortcount\endcsname by 1\relax
175   \expandafter\protected@xdef\csname glo@##2@sort\endcsname{%
176     \expandafter\glssortnumberfmt
177     {\csname glossary@##1@sortcount\endcsname}}%
178 }%
```

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

```
179 \def\@gls@setsort##1{%
180 }
```

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

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

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

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

Defined count register associated with the glossary.

```
183 \def\@gls@defsortcount##1{%
184   \expandafter\global
185   \expandafter\newcount\csname glossary@##1@sortcount\endcsname
186 }%
```

Initialise the sort key to empty.

```
187 \def\@gls@defsort##1##2{%
188   \expandafter\gdef\csname glo@##2@sort\endcsname{%
189 }%
```

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

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

Get the parent, if one exists

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

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

```
192   \ifx\@glo@parent\@empty
193   \else
194     \expandafter\@gls@setsort\expandafter{\@glo@parent}%
195   \fi
```

Set index information for this entry

```
196   \edef\@glo@type{\csname glo@##1@type\endcsname}%
197   \edef\@gls@tmp{\csname glo@##1@sort\endcsname}%
198   \ifx\@gls@tmp\@empty
199     \expandafter\global\expandafter
200     \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax
201     \expandafter\protected@xdef\csname glo@##1@sort\endcsname{%
202       \expandafter\glssortnumberfmt
203       {\csname glossary@\@glo@type @sortcount\endcsname}}%
```

```

204     \@glo@storeentry{##1}%
205     \fi
206   }%
207 }

```

`\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.)

```

208 \newcommand*{\glsdefmain}{%
209   \if@gls@docloaded
210     \newglossary[glg2]{main}{gls2}{glo2}{\glossaryname}%
211   \else
212     \newglossary{main}{gls}{glo}{\glossaryname}%
213   \fi
214 }

```

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`

```

215 \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`

```

216 \newcommand*{\acronymtype}{\glsdefaulttype}

```

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

```

217 \DeclareOptionX{nomain}{%
218   \let\glsdefaulttype\relax
219   \renewcommand*{\glsdefmain}{}%
220 }

```

`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.

```

221 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
222   \ifglsacronym
223     \renewcommand*{\@gls@do@acronymsdef}{%
224       \DeclareAcronymList{acronym}%

```

```

225     \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
226     \renewcommand*{\acronymtype}{acronym}%
227 }%
228 \else
229     \let\@gls@do@acronymsdef\relax
230 \fi
231 }

```

`\printacronyms` Define `\printacronyms` at the start of the document if acronym is set and compatibility mode isn't on and `\printacronyms` hasn't already been defined.

```

232 \AtBeginDocument{%
233     \ifglsacronym
234     \ifbool{glscompatible-3.07}%
235     {%
236     {%
237         \providecommand*{\printacronyms}[1][]{%
238             \printglossary[type=\acronymtype,#1]}%
239     }%
240 \fi
241 }

```

`@gls@do@acronymsdef` Set default value

```

242 \newcommand*{\@gls@do@acronymsdef}{}

```

`acronyms` Provide a synonym for `acronym=true` that can be passed via the document class options.

```

243 \DeclareOptionX{acronyms}{%
244     \glsacronymtrue
245 \DeclareAcronymList{acronym}%
246 }

```

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

```

247 \newcommand*{\@glsacronymlists}{}

```

`\@addtoacronymlists`

```

248 \newcommand*{\@addtoacronymlists}[1]{%
249     \ifx\@glsacronymlists\@empty
250     \protected@xdef\@glsacronymlists{#1}%
251     \else
252     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
253 \fi
254 }

```

`\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.)

```

255 \newcommand*\DeclareAcronymList}[1]{%
256   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
257 }

```

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

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

```

258 \newcommand*\glsIfListOfAcronyms}[1]{%
259   \edef\@do@gls@islistofacronyms{%
260     \noexpand\@gls@islistofacronyms{#1}{\@glsacronymlists}}%
261   \@do@gls@islistofacronyms
262 }

```

Internal command requires label and list to be expanded:

```

263 \newcommand*\@gls@islistofacronyms}[4]{%
264   \def\gls@islistofacronyms##1,#1,##2\end@gls@islistofacronyms{%
265     \def\@before{##1}\def\@after{##2}}%
266   \gls@islistofacronyms,#2,#1,\@nil\end@gls@islistofacronyms
267   \ifx\@after\@nnil

```

Not found

```

268     #4%
269   \else

```

Found

```

270     #3%
271   \fi
272 }

```

`\if@glsisacronymlist` Convenient boolean.

```

273 \newif\if@glsisacronymlist

```

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

```

274 \newcommand*\@gls@checkisacronymlist}[1]{%
275   \glsIfListOfAcronyms{#1}%
276   {\@glsisacronymlisttrue}{\@glsisacronymlistfalse}%
277 }

```

`\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.)

```

278 \newcommand*\SetAcronymLists}[1]{%
279   \renewcommand*\@glsacronymlists{#1}%
280 }

```

`acronymlists`

```

281 \define@key{glossaries.sty}{acronymlists}{%
282   \DeclareAcronymList{#1}%
283 }

```

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`

```
284 \newcommand{\glscounter}{page}
```

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

```
285 \define@key{glossaries.sty}{counter}{%
```

```
286   \renewcommand*{\glscounter}{#1}%
```

```
287 }
```

`\@gls@nohyperlist`

```
288 \newcommand*{\@gls@nohyperlist}{}%
```

`sDeclareNoHyperList`

```
289 \newcommand*{\GlsDeclareNoHyperList}[1]{%
```

```
290   \ifdefempty\@gls@nohyperlist
```

```
291   {%
```

```
292     \renewcommand*{\@gls@nohyperlist}{#1}%
```

```
293   }%
```

```
294   {%
```

```
295     \appto\@gls@nohyperlist{,#1}%
```

```
296   }%
```

```
297 }
```

`nohypertypes`

```
298 \define@key{glossaries.sty}{nohypertypes}{%
```

```
299   \GlsDeclareNoHyperList{#1}%
```

```
300 }
```

`\GlossariesWarning` Prints a warning message.

```
301 \newcommand*{\GlossariesWarning}[1]{%
```

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

```
303 }
```

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

```
304 \newcommand*{\GlossariesWarningNoLine}[1]{%
```

```
305   \PackageWarningNoLine{glossaries}{#1}%
```

```
306 }
```

Define package option to suppress warnings

```
307 \DeclareOptionX{nowarn}{%
```

```
308   \renewcommand*{\GlossariesWarning}[1]{}%
```

```
309   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
```

```
310 }
```

As from version 3.08a, the only information written to the external glossary files are the label and sort values. Therefore, now, the only sanitize option that makes sense is the one for the sort key. so the sanitize option is now deprecated and there is only a sanitizesort option.

\@gls@sanitizedesc

```

311 \newcommand*{\@gls@sanitizedesc}{%
312 }
313 %\end{macro}
314 %
315 %\begin{macro}{\glssetexpandfield}
316 %\changes{3.13a}{2013-11-05}{new}
317 %\begin{definition}
318 %\cs{glssetexpandfield}\marg{field}
319 %\end{definition}
320 % Sets field to always expand.
321 %   \begin{macrocode}
322 \newcommand*{\glssetexpandfield}[1]{%
323   \csdef{gls@assign@#1@field}##1##2{%
324     \@gls@expand@field{##1}{#1}{##2}%
325   }%
326 }
```

\glssetnoexpandfield

\glssetnoexpandfield{<field>}

Sets field to never expand.

```

327 \newcommand*{\glssetnoexpandfield}[1]{%
328   \csdef{gls@assign@#1@field}##1##2{%
329     \@gls@noexpand@field{##1}{#1}{##2}%
330   }%
331 }
```

s@assign@type@field

The type must always be expandable.

```
332 \glssetexpandfield{type}
```

s@assign@desc@field

The description is not expanded by default:

```
333 \glssetnoexpandfield{desc}
```

gn@descplural@field

```
334 \glssetnoexpandfield{descplural}
```

\@gls@sanitizename

```
335 \newcommand*{\@gls@sanitizename}{}
```

s@assign@name@field

Don't expand name by default.

```
336 \glssetnoexpandfield{name}
```


@gls@sanitizesymbol

```
337 \newcommand*{\@gls@sanitizesymbol}{}
```

assign@symbol@field Don't expand symbol by default.

```
338 \glssetnoexpandfield{symbol}
```

@symbolplural@field

```
339 \glssetnoexpandfield{symbolplural}
```

Sanitizing stuff:

\@gls@sanitizesort

```
340 \newcommand*{\@gls@sanitizesort}{%
341   \ifglssanitizesort
342     \@onelevel@sanitize\@glo@sort
343   \else
344     \fi
345 }
```

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.

```
346 \define@boolkey[gls]{sanitize}{description}[true]{%
347   \GlossariesWarning{sanitize={description} package option deprecated}%
348   \ifgls@sanitize@description
349     \glssetnoexpandfield{desc}%
350     \glssetnoexpandfield{descplural}%
351   \else
352     \glssetexpandfield{desc}%
353     \glssetexpandfield{descplural}%
354   \fi
355 }
```

```
356 \define@boolkey[gls]{sanitize}{name}[true]{%
357   \GlossariesWarning{sanitize={name} package option
358 deprecated}%
359   \ifgls@sanitize@name
360     \glssetnoexpandfield{name}%
361   \else
362     \glssetexpandfield{name}%
363   \fi
364 }
```

```
365 \define@boolkey[gls]{sanitize}{symbol}[true]{%
366   \GlossariesWarning{sanitize={symbol} package option
367 deprecated}%
368   \ifgls@sanitize@symbol
369     \glssetnoexpandfield{symbol}%
370     \glssetnoexpandfield{symbolplural}%
371   \else
```

```

372 \glssetexpandfield{symbol}%
373 \glssetexpandfield{symbolplural}%
374 \fi
375 }

```

sanitizesort

```

376 \define@boolkey{glossaries.sty}[gls]{sanitizesort}[true]{%
377 \ifglssanitizesort
378 \glssetnoexpandfield{sortvalue}%
379 \else
380 \glssetexpandfield{sortvalue}%
381 \fi
382 }

```

Default setting:

```

383 \glssanitizesorttrue
384 \glssetnoexpandfield{sortvalue}%

385 \define@choicekey{sanitization}{sort}{true,false}[true]{%
386 \setbool{glssanitizesort}{#1}%
387 \ifglssanitizesort
388 \glssetnoexpandfield{sortvalue}%
389 \else
390 \glssetexpandfield{sortvalue}%
391 \fi
392 \GlossariesWarning{sanitization={sort} package option
393 deprecated. Use sanitizesort instead}%
394 }

```

sanitize

```

395 \define@key{glossaries.sty}{sanitization}[description=true,symbol=true,
396 name=true]{%
397 \ifthenelse{\equal{#1}{none}}{%
398 {%
399 \GlossariesWarning{sanitization package option deprecated}%
400 }%
401 {%
402 \setkeys[gls]{sanitization}{#1}%
403 }%
404 }

```

\ifglstranslate As from version 3.13a, the translator package option is a choice rather than boolean option so now need to define conditional:

```

405 \newif\ifglstranslate

```

ls@nottranslatorhook

```

406 \newcommand*\@gls@nottranslatorhook{}

```

notranslate Provide a synonym for `translate=false` that can be passed via the document class.

```
407 \DeclareOptionX{notranslate}{%
408   \glstranslatefalse
409   \let\@gls@notranslatorhook\relax
410 }
```

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

```
411 \define@choicekey{glossaries.sty}{translate}[\val\nr]%
412 {true,false,babel}[true]%
413 {%
414   \ifcase\nr\relax
415     \glstranslatetrue
416   \or
417     \glstranslatefalse
418     \let\@gls@notranslatorhook\relax
419   \or
420     \glstranslatefalse
421     \def\@gls@notranslatorhook{\RequirePackage{glossaries-babel}}%
422   \fi
423 }
```

Set the default value:

```
424 \glstranslatefalse
425 \@ifpackageloaded{translator}%
426   {\glstranslatetrue}%
427   {%
428     \@ifpackageloaded{polyglossia}%
429       {\glstranslatetrue}%
430       {%
431         \@ifpackageloaded{babel}{\glstranslatetrue}{}}%
432       }%
433 }
```

indexonlyfirst Set whether to only index on first use.

```
434 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{}
435 \glsindexonlyfirstfalse
```

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

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

\@gls@setacrstyle Keep track of whether an acronym style has been set (for the benefit of `\setupglossaries`):

```
438 \newcommand*{\@gls@setacrstyle}{}%
```

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

```
439 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{}%
```

```

440 \ifbool{glsacrdescription}%
441 {}%
442 {%
443 \renewcommand*{\@gls@sanitizedesc}{}%
444 }%
445 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
446 }

```

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

```

447 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{%
448 \renewcommand*{\@gls@sanitizesymbol}{}%
449 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
450 }

```

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

```

451 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{%
452 \renewcommand*{\@gls@sanitizesymbol}{}%
453 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
454 }

```

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

```

455 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{%
456 \renewcommand*{\@gls@sanitizesymbol}{}%
457 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
458 }

```

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

```

459 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
460 \renewcommand*{\@gls@sanitizesymbol}{}%
461 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
462 }

```

shortcuts Define acronym shortcuts.

```

463 \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.

```

464 \newcommand*{\glsorder}{word}

```

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

```

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

```

order

```

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

```

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

```
468 \newif\ifglxindy
```

The default is `makeindex`:

```
469 \glxindyfalse
```

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

```
470 \DeclareOptionX{makeindex}{\glxindyfalse}
```

The `xindy` package option may have a value which in turn can be a key=value list. First define the keys for this sub-list. The boolean `glxnumbers` determines whether to automatically add the `glxnumbers` letter group.

```
471 \define@boolkey[glx]{xindy}{glxnumbers}[true]{}
```

```
472 \glx@xindy@glxnumberstrue
```

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

```
473 \def\@xdy@main@language{\language}%
```

Define key to set the language

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

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

```
475 \ifcsundef{inputencodingname}{%
```

```
476 \def\glx@codepage{}}{%
```

```
477 \def\glx@codepage{\inputencodingname}
```

```
478 }
```

Define a key to set the code page.

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

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

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

```
481 \glxindytrue
```

```
482 \setkeys[glx]{xindy}{#1}%
```

```
483 }
```

`xindygloss` Provide a synonym for `xindy` that can be passed via the document class options.

```
484 \DeclareOptionX{xindygloss}{%
```

```
485 \glxindytrue
```

```
486 }
```

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

```
487 \define@boolkey{glossaries.sty}[gls]{savewrites}[true]{%
488   \ifglssavewrites
489     \renewcommand*{\glswritefiles}{\@glswritefiles}%
490   \else
491     \let\glswritefiles\relax
492   \fi
493 }
```

Set default:

```
494 \glssavewritesfalse
495 \let\glswritefiles\relax
```

compatible-3.07

```
496 \define@boolkey{glossaries.sty}[gls]{compatible-3.07}[true]{%
497 \boolfalse{glscpatible-3.07}
```

compatible-2.07

```
498 \define@boolkey{glossaries.sty}[gls]{compatible-2.07}[true]{%
  Also set 3.07 compatibility if this option is set.
499   \ifbool{glscpatible-2.07}%
500   {%
501     \booltrue{glscpatible-3.07}%
502   }%
503   {}%
504 }
505 \boolfalse{glscpatible-2.07}
```

symbols Create a “symbols” glossary type

```
506 \newcommand{\@gls@do@symbolsdef}{}
507 \DeclareOptionX{symbols}{%
508   \renewcommand{\@gls@do@symbolsdef}{%
509     \newglossary[slg]{symbols}{sls}{slo}{\glssymbolsgroupname}%
510     \newcommand*{\printsymbols}[1] [] {\printglossary[type=symbols,#1]}%
511   }%
512 }
```

numbers Create a “symbols” glossary type

```
513 \newcommand{\@gls@do@numbersdef}{}
514 \DeclareOptionX{numbers}{%
515   \renewcommand{\@gls@do@numbersdef}{%
516     \newglossary[nlg]{numbers}{nls}{nlo}{\glsnumbersgroupname}%
517     \newcommand*{\printnumbers}[1] [] {\printglossary[type=numbers,#1]}%
518   }%
519 }
```

Process package options:

```
520 \ProcessOptionsX
```

Load backward compatibility stuff:

```
521 \RequirePackage{glossaries-compatible-307}
```

`\setupglossaries` Provide way to set options after package has been loaded. However, some options must be set before `\ProcessOptionsX`, so they have to be disabled:

```
522 \disable@keys{glossaries.sty}{compatible-2.07,%
```

```
523 xindy,xindygloss,makeindex,%
```

```
524 acronym,translate,notranslate,nolong,nosuper,notree,nostyles,nomain}
```

Now define `\setupglossaries`:

```
525 \newcommand*{\setupglossaries}[1]{%
```

```
526 \renewcommand*{\@gls@setacrstyle}{}%
```

```
527 \ifglsacrshortcuts
```

```
528 \def\@gls@setupshortcuts{\glsacrshortcutstrue}%
```

```
529 \else
```

```
530 \def\@gls@setupshortcuts{%
```

```
531 \ifglsacrshortcuts
```

```
532 \DefineAcronymSynonyms
```

```
533 \fi
```

```
534 }%
```

```
535 \fi
```

```
536 \glsacrshortcutsfalse
```

```
537 \let\@gls@do@numbersdef\relax
```

```
538 \let\@gls@do@symbolssdef\relax
```

```
539 \let\@gls@do@acronymsdef\relax
```

```
540 \setkeys{glossaries.sty}{#1}%
```

```
541 \@gls@setacrstyle
```

```
542 \@gls@setupshortcuts
```

```
543 \@gls@do@acronymsdef
```

```
544 \@gls@do@numbersdef
```

```
545 \@gls@do@symbolssdef
```

```
546 }
```

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

```
547 \ifglstranslate
```

```
548 \@ifpackageloaded{polyglossia}%
```

```
549 {%
```

polyglossia fakes babel so need to check for polyglossia first.

```
550 }%
```

```
551 {%
```

```
552 \@ifpackageloaded{babel}%
```

```
553 {%
```

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

```
555 {%
```

```
556 \RequirePackage{translator}%
```

```

557         }%
558     {}%
559     }%
560     {}
561 }
562 \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}`).

```

563 \ifthenelse{\equal{\glscounter}{section}}{%
564 {%
565     \ifcsundef{chapter}{}%
566     {%
567         \let\@gls@old@chapter\@chapter
568         \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
569             \ifcsundef{hyperdef}{\hyperdef{section}{\thesection}}}%
570     }%
571 }%
572 {}

```

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

```

573 \newcommand*\@gls@onlypremakeg{}

```

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

```

574 \newcommand*\@onlypremakeg[1]{%
575     \ifx\@gls@onlypremakeg\@empty
576         \def\@gls@onlypremakeg{#1}%
577     \else
578         \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
579         \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
580     \fi
581 }

```

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

```

582 \newcommand*\@disable@onlypremakeg{%
583     \@for\@thiscs:=\@gls@onlypremakeg\do{%
584         \expandafter\@disable@premakecs\@thiscs%
585     }}

```


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

```
586 \newcommand*{\@disable@premakecs}[1]{%
587   \def#1{\PackageError{glossaries}{\string#1\space may only be
588     used before \string\makeglossaries}{You can't use
589     \string#1\space after \string\makeglossaries}}%
590 }
```

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`

```
591 \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`

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

`\glstocctitle`

Sets the TOC title for the given glossary.

```
593 \newcommand*{\glstocctitle}[1]{%
594 \def\glossarytocctitle{\csname @glotype@#1@title\endcsname}}
```

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

`\entryname`

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

`\descriptionname`

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

`\symbolname`

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

`\pagelistname`

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

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

`glssymbolsgroupname`

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

glsnumbersgroupname

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

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

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

\seename

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

\andname

```
603 \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.)

```
604 \newcommand*{\addglossarytocaptions}[1]{%
605   \ifcsundef{captions#1}{}%
606   {%
607     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
608     \expandafter\toks@\expandafter{\@gls@tmp
609       \renewcommand*{\glossaryname}{\translate{Glossary}}}%
610     }%
611     \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
612   }%
613 }
614 \ifglstranslate
```

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

```
615 \@ifpackageloaded{translator}{%
616   \usedictionary{glossaries-dictionary}%
617   \addglossarytocaptions{portuges}%
618   \addglossarytocaptions{portuguese}%
619   \addglossarytocaptions{brazil}%
620   \addglossarytocaptions{brazilian}%
621   \addglossarytocaptions{danish}%
622   \addglossarytocaptions{dutch}%
623   \addglossarytocaptions{afrikaans}%
624   \addglossarytocaptions{english}%
625   \addglossarytocaptions{UKenglish}%
626   \addglossarytocaptions{USenglish}%
627   \addglossarytocaptions{american}%
628   \addglossarytocaptions{australian}%
629   \addglossarytocaptions{british}%
630   \addglossarytocaptions{canadian}%
```

```

631 \addglossarytocaptions{newzealand}%
632 \addglossarytocaptions{french}%
633 \addglossarytocaptions{frenchb}%
634 \addglossarytocaptions{français}%
635 \addglossarytocaptions{acadian}%
636 \addglossarytocaptions{canadien}%
637 \addglossarytocaptions{german}%
638 \addglossarytocaptions{germanb}%
639 \addglossarytocaptions{austrian}%
640 \addglossarytocaptions{naustrian}%
641 \addglossarytocaptions{ngerman}%
642 \addglossarytocaptions{irish}%
643 \addglossarytocaptions{italian}%
644 \addglossarytocaptions{magyar}%
645 \addglossarytocaptions{hungarian}%
646 \addglossarytocaptions{polish}%
647 \addglossarytocaptions{spanish}%
648 \renewcommand*{\glssettoctitle}[1]{%
649 \ifthenelse{\equal{#1}{main}}{%
650 \translatelet{\glossarytoctitle}{Glossary}}{%
651 \ifthenelse{\equal{#1}{acronym}}{%
652 \translatelet{\glossarytoctitle}{Acronyms}}{%
653 \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
654 \renewcommand*{\glossaryname}{\translate{Glossary}}%
655 \renewcommand*{\acronymname}{\translate{Acronyms}}%
656 \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
657 \renewcommand*{\descriptionname}{%
658 \translate{Description (glossaries)}}%
659 \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
660 \renewcommand*{\pagelistname}{%
661 \translate{Page List (glossaries)}}%
662 \renewcommand*{\glssymbolsgroupname}{%
663 \translate{Symbols (glossaries)}}%
664 \renewcommand*{\glsnumbersgroupname}{%
665 \translate{Numbers (glossaries)}}%
666 }{%

667 \@ifpackageloaded{polyglossia}%
668 {\RequirePackage{glossaries-polyglossia}}%
669 {%
670 \@ifpackageloaded{babel}{%
671 \RequirePackage{glossaries-babel}}{%
672 }}
673 \else

674 \@gls@notranslatorhook
675 \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.

```
676 \DeclareRobustCommand*\nopostdesc{}\}
```

`\nopostdesc` Suppress next description terminator.

```
677 \newcommand*\nopostdesc{%
678   \let\org@glspostdescription\glspostdescription
679   \def\glspostdescription{%
680     \let\glspostdescription\org@glspostdescription}%
681 }
```

`\no@post@desc` Used for comparison purposes.

```
682 \newcommand*\no@post@desc{}\nopostdesc}
```

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

```
683 \newcommand{\glspar}{\par}
```

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

```
684 \ifglsxindy
685   \newcommand{\setStyleFile}[1]{%
686     \renewcommand{\istfilename}{#1.xdy}}
687 \else
688   \newcommand{\setStyleFile}[1]{%
689     \renewcommand{\istfilename}{#1.ist}}
690 \fi
```

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

```
691 \@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 re-defining 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`

```
692 \ifglsxindy
693   \def\istfilename{\jobname.xdy}
694 \else
695   \def\istfilename{\jobname.ist}
696 \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`

```
697 \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`

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

`\glsSetCompositor` Sets the compositor.

```
699 \newcommand*\glsSetCompositor[1]{%
700   \renewcommand*\glscompositor}{#1}}
    Only use before \makeglossaries
701 \@onlypremakeg\glsSetCompositor
```

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

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

`glsSetAlphaCompositor` Sets the alpha compositor.

```
703 \ifglsxindy
704   \newcommand*\glsSetAlphaCompositor[1]{%
705     \renewcommand*\@glsAlphacompositor{#1}}
706 \else
707   \newcommand*\glsSetAlphaCompositor[1]{%
708     \glsnoxindywarning\glsSetAlphaCompositor}
709 \fi
    Can only be used before \makeglossaries
710 \@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.

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

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

```
712 \newcommand*\glsSetSuffixF[1]{%
713   \renewcommand*\gls@suffixF}{#1}}
    Only has an effect when used before \makeglossaries
714 \@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.

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

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

```

716 \newcommand*\glsSetSuffixFF}[1]{%
717   \renewcommand*\gls@suffixFF{#1}%
718 }
```

`\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”.

```

719 \ifcsundef{hyperlink}%
720 {%
721   \newcommand*\glsnumberformat}[1]{#1}%
722 }%
723 {%
724   \newcommand*\glsnumberformat}[1]{\glshypernumber{#1}}%
725 }
```

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`

```

726 \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`

```

727 \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`

```

728 \newcommand*\glossarypreamble{%
729   \csuse{@glossarypreamble@\currentglossary}%
730 }
```

`\setglossarypreamble` `\setglossarypreamble[<type>]{<text>}`

Code provided by Michael Pock.

```

731 \newcommand{\setglossarypreamble}[2][\glsdefaulttype]{%
732   \ifglossaryexists{#1}{%
733     \csgdef{@glossarypreamble@#1}{#2}%
734   }{%
735     \GlossariesWarning{%
736       Glossary ‘#1’ is not defined%
737     }%
738   }%
739 }

```

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

```

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

```

`\glossarypostamble`

```

740 \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`.

```

741 \newcommand*{\glossarysection}[2][\@gls@title]{%
742   \def\@gls@title{#2}%
743   \ifcsundef{phantomsection}%
744   {%
745     \@glossarysection{#1}{#2}%
746   }%
747   {%
748     \p@glossarysection{#1}{#2}%
749   }%

750   \glsglossarymark{\glossarytoctitle}%
751 }

```

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

```

752 \ifcsundef{glossarymark}%
753 {%
754   \newcommand{\glsglossarymark}[1]{\glossarymark{#1}}

```

```

755 }%
756 {%
757   \@ifclassloaded{memoir}
758   {%
759     \newcommand{\glsglossarymark}[1]{%
760       \ifglsucmark
761         \markboth{\memUHead{#1}}{\memUHead{#1}}%
762       \else
763         \markboth{#1}{#1}%
764       \fi
765     }
766   }%
767   {%
768     \newcommand{\glsglossarymark}[1]{%
769       \ifglsucmark
770         \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
771       \else
772         \@mkboth{#1}{#1}%
773       \fi
774     }
775   }
776 }

```

`\glossarymark` Provided for backward compatibility:

```

777 \providecommand{\glossarymark}[1]{%
778   \ifglsucmark
779     \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
780   \else
781     \@mkboth{#1}{#1}%
782   \fi
783 }

```

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`

```

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

```

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

`\@glossarysection`

```

786 \newcommand*\@glossarysection[2]{%
787   \ifx\@glossarysecstar\@empty
788     \csname\@glossarysec\endcsname{#2}%

```



```

789 \else
790   \csname\@glossarysec\endcsname*{#2}%
791   \@gls@toc{#1}{\@glossarysec}%
792 \fi
793 \@glossaryseclabel
794 }

```

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

`\@p@glossarysection`

```

795 \newcommand*{\@p@glossarysection}[2]{%
796   \gls@clearpage
797   \phantomsection
798   \ifx\@glossarysecstar\@empty
799     \csname\@glossarysec\endcsname*{#2}%
800   \else
801     \@gls@toc{#1}{\@glossarysec}%
802     \csname\@glossarysec\endcsname*{#2}%
803   \fi
804   \@glossaryseclabel
805 }

```

`\gls@docclearpage` The `\gls@docclearpage` 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.

```

806 \newcommand*{\gls@docclearpage}{%
807   \ifthenelse{\equal{\@glossarysec}{chapter}}{%
808     {%
809       \ifcsundef{cleardoublepage}%
810       {%
811         \clearpage
812       }%
813     }%
814     \ifcsdef{if@openright}%
815     {%
816       \if@openright
817         \cleardoublepage
818       \else
819         \clearpage
820       \fi
821     }%
822     {%
823       \cleardoublepage
824     }%
825   }%
826 }%

```

```

827  {}%
828 }

```

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

```

829 \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`

```

830 \newcommand*{\@gls@toc}[2]{%
831   \ifglstoc
832     \ifglsnumberline
833       \addcontentsline{toc}{#2}{\numberline{#1}}%
834     \else
835       \addcontentsline{toc}{#2}{#1}%
836     \fi
837   \fi
838 }

```

1.4 Xindy

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

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

```

839 \newcommand*{\glsnoxindywarning}[1]{%
840   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
841 }

```

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

```

842 \ifglsxindy
843   \edef\@xdyattributes{\string"default\string"}%
844 \fi

```

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

```

845 \ifglsxindy
846   \edef\@xdyattributelist{}%
847 \fi

```

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

```

848 \ifglsxindy
849   \def\@xdylocref{}
850 \fi

```

\@gls@ifinlist

```

851 \newcommand*{\@gls@ifinlist}[4]{%
852   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
853     \def\@gls@listsuffix{##2}%
854     \ifx\@gls@listsuffix\@empty
855       #4%
856     \else
857       #3%
858     \fi
859   }%
860   \@do@ifinlist,#2,#1,\end@do@ifinlist
861 }

```

\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.

```

862 \ifglxsindy
863   \newcommand*{\@xdycounters}{\@glscounter}
864   \newcommand*\GlsAddXdyCounters[1]{%
865     \@for\@gls@ctr:=#1\do{%
      Check if already in list before adding.
866       \edef\@do@addcounter{%
867         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
868         {%
869           \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
870             \noexpand\@gls@ctr}%
871         }%
872       }%
873       \@do@addcounter
874     }
875   }

```

Only has an effect before \writeist:

```

876   \@onlypremakeg\GlsAddXdyCounters
877 \else
878   \newcommand*\GlsAddXdyCounters[1]{%
879     \glsnosindywarning\GlsAddXdyAttribute
880   }
881 \fi

```

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

```

882 \newcommand*\@disabled@glssaddxdycounters{%
883   \PackageError{glossaries}{\string\GlsAddXdyCounters\space
884     can't be used after \string\GlsAddXdyAttribute}{Move all
885     occurrences of \string\GlsAddXdyCounters\space before the first
886     instance of \string\GlsAddXdyAttribute}%
887 }

```

`\GlsAddXdyAttribute` Adds an attribute.

888 `\ifglxindy`

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

889 `\newcommand*\@glsaddxdyattribute[2]{%`

Add to xindy attribute list

890 `\edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J`
891 `\string"#2#1\string"}%`

Add to xindy markup location.

892 `\expandafter\toks@\expandafter{\@xdylocref}%`
893 `\edef\@xdylocref{\the\toks@ ^^J%`
894 `(markup-locref`
895 `:open \string"\string~n%`
896 `\expandafter\string\csname glsX#2X#1\endcsname`
897 `\string" ^^J`
898 `:close \string"\string" ^^J`
899 `:attr \string"#2#1\string"))}%`

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

900 `\expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%`
901 `\setentrycounter{##1}{#2}\csname #1\endcsname{##2}%`
902 `}%`
903 `}`

High-level command:

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

Add to comma-separated attribute list

905 `\ifx\@xdyattributelist\@empty`
906 `\edef\@xdyattributelist{#1}%`
907 `\else`
908 `\edef\@xdyattributelist{\@xdyattributelist,#1}%`
909 `\fi`

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

910 `\@for\@this@counter:=\@xdycounters\do{%`
911 `\protected@edef\gls@do@addxdyattribute{%`
912 `\noexpand\@glsaddxdyattribute{#1}{\@this@counter}%`
913 `}`
914 `\gls@do@addxdyattribute`
915 `}%`

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

916 `\let\GlsAddXdyCounters\@disabled@glsaddxdycounters`
917 `}`

Only has an effect before `\writeist`:

918 `\@onlypremakeg\GlsAddXdyAttribute`
919 `\else`
920 `\newcommand*\GlsAddXdyAttribute[1]{%`

```

921 \glsnoxywarning\GlsAddXdyAttribute}
922 \fi

```

redefinedattributes Add known attributes for all defined counters

```

923 \ifglxindy
924 \newcommand*{\@gls@addpredefinedattributes}{%
925   \GlsAddXdyAttribute{glsnumberformat}
926   \GlsAddXdyAttribute{textrm}
927   \GlsAddXdyAttribute{textsf}
928   \GlsAddXdyAttribute{texttt}
929   \GlsAddXdyAttribute{textbf}
930   \GlsAddXdyAttribute{textmd}
931   \GlsAddXdyAttribute{textit}
932   \GlsAddXdyAttribute{textup}
933   \GlsAddXdyAttribute{textsl}
934   \GlsAddXdyAttribute{textsc}
935   \GlsAddXdyAttribute{emph}
936   \GlsAddXdyAttribute{glshypernumber}
937   \GlsAddXdyAttribute{hyperbm}
938   \GlsAddXdyAttribute{hypersf}
939   \GlsAddXdyAttribute{hypertt}
940   \GlsAddXdyAttribute{hyperbm}
941   \GlsAddXdyAttribute{hypermd}
942   \GlsAddXdyAttribute{hyperit}
943   \GlsAddXdyAttribute{hyperup}
944   \GlsAddXdyAttribute{hypersl}
945   \GlsAddXdyAttribute{hypersc}
946   \GlsAddXdyAttribute{hyperemph}
947 }
948 \else
949   \let\@gls@addpredefinedattributes\relax
950 \fi

```

\@xdyuseralphabets List of additional alphabets

```

951 \def\@xdyuseralphabets{}

```

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

```

952 \ifglxindy
953   \newcommand*{\GlsAddXdyAlphabet}[2]{%
954     \edef\@xdyuseralphabets{%
955       \@xdyuseralphabets ^^J
956       (define-alphabet "#1" (#2))}}
957 \else
958   \newcommand*{\GlsAddXdyAlphabet}[2]{%
959     \glsnoxywarning\GlsAddXdyAlphabet}
960 \fi

```

This code is only required for xindy:

961 \ifglxindy

ls@xdy@locationlist List of predefined location names.

```
962 \newcommand*{\@gls@xdy@locationlist}{%
963     roman-page-numbers,%
964     Roman-page-numbers,%
965     arabic-page-numbers,%
966     alpha-page-numbers,%
967     Alpha-page-numbers,%
968     Appendix-page-numbers,%
969     arabic-section-numbers%
970 }
```

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.

```
971 \protected@edef\@gls@roman{\@roman{0}\string"
972     \string"roman-numbers-lowercase\string" :sep \string"}}%
973 \@onelevel@sanitize\@gls@roman
974 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
975     :sep \string"}%
976 \@onelevel@sanitize\@tmp
977 \ifx\@tmp\@gls@roman
978     \expandafter
979     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
980         \string"roman-numbers-lowercase\string"%
981     }%
982 \else
983     \expandafter
984     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
985         :sep \string"\@gls@roman\string"%
986     }%
987 \fi
```

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

```
988 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
989     \string"roman-numbers-uppercase\string"%
990 }
```

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

```
991 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
992     \string"arabic-numbers\string"%
993 }
```

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

```

994 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
995 \string"alpha\string"%
996 }%

@Alpha-page-numbers Upper case alphabetical (A, B, ...).
997 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
998 \string"ALPHA\string"%
999 }%

appendix-page-numbers Appendix style locations (e.g. A-1, A-2, ..., B-1, B-2, ...). The separator is given
by \@glsAlphacompositor.
1000 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
1001 \string"ALPHA\string"
1002 :sep \string"\@glsAlphacompositor\string"
1003 \string"arabic-numbers\string"%
1004 }

arabic-section-numbers Section number style locations (e.g. 1.1, 1.2, ...). The compositor is given by
\glscompositor.
1005 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
1006 \string"arabic-numbers\string"
1007 :sep \string"\glscompositor\string"
1008 \string"arabic-numbers\string"%
1009 }%

xdyuserlocationdefs List of additional location definitions (separated by ^^J)
1010 \def\@xdyuserlocationdefs{}

xdyuserlocationnames List of additional user location names
1011 \def\@xdyuserlocationnames{}

End of xindy-only block:
1012 \fi

\GlsAddXdyLocation \GlsAddXdyLocation[<prefix-loc>]{<name>}{<definition>} Define a new lo-
cation called <name>. The definition must use xindy syntax. (Note that this
doesn't check to see if the location is already defined. That is left to xindy to
complain about.)
1013 \ifglsxindy
1014 \newcommand*\GlsAddXdyLocation{3}[]{}%
1015 \def\@gls@tmp{#1}%
1016 \ifx\@gls@tmp\@empty
1017 \edef\@xdyuserlocationdefs{%
1018 \xdyuserlocationdefs ^^J%
1019 (define-location-class \string"#2\string"^^J\space\space
1020 \space(:sep \string"{}\glsopenbrace\string" #3
1021 :sep \string"\glsclosebrace\string"))

```

```

1022     }%
1023   \else
1024     \edef\@xdyuserlocationdefs{%
1025       \@xdyuserlocationdefs ^^J%
1026       (define-location-class \string"#2\string"^^J\space\space
1027         \space(:sep "\glsoopenbrace"
1028           #1
1029           :sep "\glsclosebrace\glsoopenbrace" #3
1030           :sep "\glsclosebrace"))
1031     }%
1032   \fi
1033   \edef\@xdyuserlocationnames{%
1034     \@xdyuserlocationnames^^J\space\space\space
1035     \string"#1\string"}%
1036 }

```

Only has an effect before \writeist:

```

1037 \@onlypremakeg\GlsAddXdyLocation
1038 \else
1039   \newcommand*\GlsAddXdyLocation}[2]{%
1040     \glsnnoxindywarning\GlsAddXdyLocation}
1041 \fi

```

ylocationclassorder Define location class order

```

1042 \ifglxindy
1043   \edef\@xdylocationclassorder{^^J\space\space\space
1044     \string"roman-page-numbers\string"^^J\space\space\space
1045     \string"arabic-page-numbers\string"^^J\space\space\space
1046     \string"arabic-section-numbers\string"^^J\space\space\space
1047     \string"alpha-page-numbers\string"^^J\space\space\space
1048     \string"Roman-page-numbers\string"^^J\space\space\space
1049     \string"Alpha-page-numbers\string"^^J\space\space\space
1050     \string"Appendix-page-numbers\string"
1051     \@xdyuserlocationnames^^J\space\space\space
1052     \string"see\string"
1053   }
1054 \fi

```

Change the location order.

yLocationClassOrder

```

1055 \ifglxindy
1056   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1057     \def\@xdylocationclassorder{#1}}
1058 \else
1059   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1060     \glsnnoxindywarning\GlsSetXdyLocationClassOrder}
1061 \fi

```

\@xdysortrules Define sort rules


```

1062 \ifglxindy
1063   \def\@xdysortrules{}
1064 \fi

```

`\GlsAddSortRule` Add a sort rule

```

1065 \ifglxindy
1066   \newcommand*\GlsAddSortRule[2]{%
1067     \expandafter\toks@\expandafter{\@xdysortrules}%
1068     \protected@edef\@xdysortrules{\the\toks@ ^^J
1069       (sort-rule \string"#1\string" \string"#2\string"))}%
1070   }
1071 \else
1072   \newcommand*\GlsAddSortRule[2]{%
1073     \glsnxindywarning\GlsAddSortRule}
1074 \fi

```

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

```

1075 \ifglxindy
1076   \def\@xdyrequiredstyles{tex}
1077 \fi

```

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

```

1078 \ifglxindy
1079   \newcommand*\GlsAddXdyStyle[1]{%
1080     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
1081 \else
1082   \newcommand*\GlsAddXdyStyle[1]{%
1083     \glsnxindywarning\GlsAddXdyStyle}
1084 \fi

```

`\GlsSetXdyStyles` Reset the list of required styles

```

1085 \ifglxindy
1086   \newcommand*\GlsSetXdyStyles[1]{%
1087     \edef\@xdyrequiredstyles{#1}}
1088 \else
1089   \newcommand*\GlsSetXdyStyles[1]{%
1090     \glsnxindywarning\GlsSetXdyStyles}
1091 \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).

```

1092 \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.

```
1093 \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.

```
1094 \ifglxindy
1095   \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
1096     \ifglossaryexists{#1}{%
1097       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
1098     }{%
1099       \PackageError{glossaries}{Can't set language type for
1100         glossary type '#1' --- no such glossary}{%
1101         You have specified a glossary type that doesn't exist}}
1102 \else
1103   \newcommand*\GlsSetXdyLanguage[2][]{%
1104     \glsnoxywarning\GlsSetXdyLanguage}
1105 \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.

```
1106 \def\@gls@codepage#1#2{}
```

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

```
1107 \ifglxindy
1108   \newcommand*\GlsSetXdyCodePage[1]{%
1109     \renewcommand*\@gls@codepage{#1}%
1110   }
1111   Suggested by egreg:
1112   \AtBeginDocument{%
1113     \ifx\@gls@codepage\@empty
1114       \ifpackageoption{fontspec}{\def\@gls@codepage{utf8}}{}%
1115     \fi
1116   \else
1117     \newcommand*\GlsSetXdyCodePage[1]{%
1118       \glsnoxywarning\GlsSetXdyCodePage}
1119 \fi
```

`\@xdylettergroups` Store letter group definitions.

```
1120 \ifglxindy
1121   \ifglxindy@glslnumbers
1122     \def\@xdylettergroups{(define-letter-group
1123       \string\glslnumbers\string"^^J\space\space\space
```

```

1124      :prefixes (\string"0\string" \string"1\string"
1125      \string"2\string" \string"3\string" \string"4\string"
1126      \string"5\string" \string"6\string" \string"7\string"
1127      \string"8\string" \string"9\string")^^J\space\space\space
1128      :before \string"@glsfirstletter\string"))}
1129  \else
1130    \def\xdylettergroups{}
1131  \fi
1132 \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.

```

1133 \newcommand*\GlsAddLetterGroup[2]{%
1134   \expandafter\toks@\expandafter{\@xdylettergroups}%
1135   \protected@edef\xdylettergroups{\the\toks@^^J%
1136   (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
1137 }%

```

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.

```

1138 \newcommand*\forallglossaries[3][\@glo@types]{%
1139   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
1140 }

```

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

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

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

```

1141 \newcommand*\forglsentries[3][\glsdefaulttype]{%
1142   \edef\@glo@list{\csname glolist@#1\endcsname}%
1143   \@for#2:=\@glo@list\do
1144   {%
1145     \ifdefempty{#2}{#3}%
1146   }%
1147 }

```

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

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

```

1148 \newcommand*{\forallglsentries}[3][\@glo@types]{%
1149   \expandafter\forallglossaries\expandafter[#1]{\@@this@glo@}%
1150   {%
1151     \forglsentries[\@@this@glo@]{#2}{#3}%
1152   }%
1153 }

```

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

```

1154 \newcommand{\ifglossaryexists}[3]{%
1155   \ifcsundef{@glotype@#1@out}{#3}{#2}%
1156 }

```

$$\text{\ifglentryexists}\{\langle label \rangle\}\{\langle true\text{ text} \rangle\}\{\langle false\text{ text} \rangle\}$$

```

1157 \newcommand{\ifglentryexists}[3]{%
1158   \ifcsundef{glo@#1@name}{#3}{#2}%
1159 }

```

$$\text{\ifglused{\langle label \rangle}\{\langle true text \rangle\}\{\langle false text \rangle\}}$$

```
1160 \newcommand*{\ifglsused}[3]{\ifbool{glo@#1@flag}{#2}{#3}}
```

$$\backslash\mathrm{glsdoifexists}\quad\backslash\mathrm{glsdoifexists}\{\langle\mathit{label}\rangle\}\{\langle\mathit{code}\rangle\}$$

```

1161 \newcommand{\glsdoifexists}[2]{%
1162   \ifglsentryexists{#1}{#2}{%
1163     \PackageError{glossaries}{Glossary entry ‘#1’ has not been
1164     defined}{You need to define a glossary entry before you
1165     can use it.}}%
1166 }

```

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

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

```
1167 \newcommand{\glsdoifnoexists}[2]{%
1168   \ifglstryexists{#1}{%
1169     \PackageError{glossaries}{Glossary entry ‘#1’ has already
1170       been defined}{#2}%
1171 }
```

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

```
1172 \newcommand{\ifglshaschildren}[3]{%
1173   \glsdoifexists{#1}%
1174   {%
1175     \def\do@glshaschildren{#3}%
1176     \expandafter\forlstryentries\expandafter[\csname glo@#1@type\endcsname]
1177     {\glo@label}%
1178     {%
1179       \letcs\glo@parent{\glo@\glo@label @parent}%
1180       \ifthenelse{\equal{#1}{\glo@parent}}{%
1181         {%
1182           \def\do@glshaschildren{#2}%
1183           \@endfortrue
1184         }%
1185       }%
1186     }%
1187     \do@glshaschildren
1188   }%
1189 }
```

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

```
1190 \newcommand{\ifglshasparent}[3]{%
1191   \glsdoifexists{#1}%
1192   {%
1193     \ifcsemtty{\glo@#1@parent}{#3}{#2}%
1194   }%
1195 }
```

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

```
1196 \newcommand*{\ifglshasdesc}[3]{%
1197   \ifcsemtty{\glo@#1@desc}%
1198   {#3}%
1199   {#2}%
1200 }
```

`\ifglsdscsuppressed` `\ifglsdscsuppressed{<label>}{<true part>}{<false part>}` Does *<true part>* if the description is just `\nopostdesc` otherwise does *<false part>*.

```
1201 \newcommand*{\ifglsdscsuppressed}[3]{%
1202   \ifcsequal{\glos@#1@desc}{@no@post@desc}%
1203 }
```

```

1203   {#2}%
1204   {#3}%
1205 }

\ifglshassymbol \ifglshassymbol{<label>}{<true part>}{<false part>}
1206 \newcommand*{\ifglshassymbol}[3]{%
1207   \ifcempty{glo@#1@symbol}%
1208   {#3}%
1209   {%
1210     \expandafter\ifx\csname glo@#1@symbol\endcsname\@gls@default@value
1211     #3%
1212   \else
1213     #2%
1214   \fi
1215 }%
1216 }

\ifglshaslong \ifglshaslong{<label>}{<true part>}{<false part>}
1217 \newcommand*{\ifglshaslong}[3]{%
1218   \ifcempty{glo@#1@long}%
1219   {#3}%
1220   {%
1221     \expandafter\ifx\csname glo@#1@long\endcsname\@gls@default@value
1222     #3%
1223   \else
1224     #2%
1225   \fi
1226 }%
1227 }

\ifglshasshort \ifglshasshort{<label>}{<true part>}{<false part>}
1228 \newcommand*{\ifglshasshort}[3]{%
1229   \ifcempty{glo@#1@short}%
1230   {#3}%
1231   {%
1232     \expandafter\ifx\csname glo@#1@short\endcsname\@gls@default@value
1233     #3%
1234   \else
1235     #2%
1236   \fi
1237 }%
1238 }

```

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

```
1239 \newcommand*{\@glo@types}{,}
```

provide@newglossary If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```
1240 \newcommand*\@gls@provide@newglossary{%
```

```
1241   \protected@write\@auxout{}\string\providecommand\string\@newglossary[4]{}}%
```

Only need to do this once.

```
1242   \let\@gls@provide@newglossary\relax
```

```
1243 }
```

\defglsentryfmt Allow different glossaries to have different display styles.

```
1244 \newcommand*\defglsentryfmt[2][\glsdefaulttype]{%
```

```
1245   \csgdef{gls@#1@entryfmt}{#2}%
```

```
1246 }
```

\gls@doentryfmt

```
1247 \newcommand*\gls@doentryfmt[1]{\csuse{gls@#1@entryfmt}}
```

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

```
1248 \newcommand*\newglossary[5][glg]{%
```

```
1249   \ifglossaryexists{#2}%
```

```
1250   {%
```

```
1251     \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
```

```
1252     You can't define a new glossary called ‘#2’ because it already
```

```
1253     exists}%
```

```
1254   }%
```

```
1255   {%
```

Check if default has been set

```
1256   \ifundef\glsdefaulttype
```

```
1257   {%
```

```

1258 \gdef\glsdefaultttype{#2}%
1259 }{}%

```

Add this to the list of glossary types:

```

1260 \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.

```

1261 \expandafter\gdef\csname glolist@#2\endcsname{,}%

```

Store details of this new glossary type:

```

1262 \expandafter\def\csname @glotype@#2@in\endcsname{#3}%
1263 \expandafter\def\csname @glotype@#2@out\endcsname{#4}%
1264 \expandafter\def\csname @glotype@#2@title\endcsname{#5}%

1265 \@gls@provide@newglossary
1266 \protected@write\@auxout{}\string\@newglossary{#2}{#1}{#3}{#4}}%

```

How to display this entry in the document text (uses \glsentry by default). This can be redefined by the user later if required (see \defglsentry). This may already have been defined if this has been specified as a list of acronyms.

```

1267 \ifcsundef{gls@#2@entryfmt}%
1268 {%
1269 \defglsentryfmt[#2]{\glsentryfmt}%
1270 }%
1271 {}%

```

Define sort counter if required:

```

1272 \@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.)

```

1273 \@ifnextchar[{\@gls@setcounter{#2}}%
1274 {\@gls@setcounter{#2}[\glscounter]}}%
1275 }

```

\altnewglossary

```

1276 \newcommand*\altnewglossary}[3]{%
1277 \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1278 }

```

Only define new glossaries in the preamble:

```

1279 \@onlypreamble{\newglossary}

```

Only define new glossaries before \makeglossaries

```

1280 \@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 \TeX , `\@newglossary` simply ignores its arguments.

`\@newglossary`

```
1281 \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`

```
1282 \def\@gls@setcounter#1[#2]{%
1283   \expandafter\def\csname @gls@#1@counter\endcsname{#2}%

```

Add counter to xindy list, if not already added:

```
1284   \ifglsxindy
1285     \GlsAddXdyCounters{#2}%
1286   \fi
1287 }

```

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

`\@gls@getcounter`

```
1288 \newcommand*{\@gls@getcounter}[1]{%
1289   \csname @gls@#1@counter\endcsname
1290 }

```

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

```
1291 \glsdefmain

```

Define the “acronym” glossaries if required.

```
1292 \@gls@do@acronymsdef

```

Define the “symbols” and “numbers” glossaries if required.

```
1293 \@gls@do@symbolsdef
1294 \@gls@do@numbersdef

```

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.

```
1295 \define@key{glossentry}{name}{%
1296 \def\@glo@name{#1}%
1297 }
```

description The description key is usually only used in the glossary, but can be made to appear in the text by redefining `\glsentryfmt` or using `\defglsentryfmt`. The description key is required when defining a new glossary entry. If a long description is required, use `\longnewglossaryentry` instead of `\newglossaryentry`.

```
1298 \define@key{glossentry}{description}{%
1299 \def\@glo@desc{#1}%
1300 }
```

descriptionplural

```
1301 \define@key{glossentry}{descriptionplural}{%
1302 \def\@glo@descplural{#1}%
1303 }
```

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>`.

```
1304 \define@key{glossentry}{sort}{%
1305 \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.

```
1306 \define@key{glossentry}{text}{%
1307 \def\@glo@text{#1}%
1308 }
```

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.

```
1309 \define@key{glossentry}{plural}{%
1310 \def\@glo@plural{#1}%
1311 }
```

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.

```
1312 \define@key{glossentry}{first}{%
1313 \def\@glo@first{#1}%
1314 }
```

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.

```
1315 \define@key{glossentry}{firstplural}{%
1316 \def\@glo@firstplural{#1}%
1317 }
```

\@gls@default@value

```
1318 \newcommand*{\@gls@default@value}{\relax}
```

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 `\glossentry`. If you want this value to appear in the text when the term is used by commands like `\gls`, you will need to change `\glsentryfmt` (or use for `\defglsentryfmt` individual glossaries).

```
1319 \define@key{glossentry}{symbol}{%
1320 \def\@glo@symbol{#1}%
1321 }
```

symbolplural

```
1322 \define@key{glossentry}{symbolplural}{%
1323 \def\@glo@symbolplural{#1}%
1324 }
```

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

```
1325 \define@key{glossentry}{type}{%
1326 \def\@glo@type{#1}}
```

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

```
1327 \define@key{glossentry}{counter}{%
1328 \ifcsundef{c@#1}%
1329 {%
1330 \PackageError{glossaries}%
1331 {There is no counter called ‘#1’}%
1332 {%
1333 The counter key should have the name of a valid counter
1334 as its value%
1335 }%
1336 }%
1337 {%
1338 \def\@glo@counter{#1}%
1339 }%
1340 }
```

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

```
1341 \define@key{glossentry}{see}{%
1342   \gls@checkseeallowed
1343   \def\@glo@see{#1}%
1344   \@glo@seeautonumberlist
1345 }
```

`gls@checkseeallowed`

```
1346 \newcommand*{\gls@checkseeallowed}{%
1347   \PackageError{glossaries}%
1348   {'see' key may only be used after \string\makeglossaries}%
1349   {You must use \string\makeglossaries\space before defining
1350    any entries that have a 'see' key}%
1351 }
```

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

```
1352 \define@key{glossentry}{parent}{%
1353   \def\@glo@parent{#1}}
```

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

```
1354 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1355   \ifcase\nr\relax
1356     \def\@glo@prefix{\glsnonextpages}%
1357   \else
1358     \def\@glo@prefix{\glsnextpages}%
1359   \fi
1360 }
```

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

user1

```
1361 \define@key{glossentry}{user1}{%
1362   \def\@glo@useri{#1}%
1363 }
```

user2

```
1364 \define@key{glossentry}{user2}{%
1365   \def\@glo@userii{#1}%
1366 }
```

user3

```
1367 \define@key{glossentry}{user3}{%
1368   \def\@glo@useriii{#1}%
1369 }
```

user4

```
1370 \define@key{glossentry}{user4}{%
1371   \def\@glo@useriv{#1}%
1372 }
```

user5

```
1373 \define@key{glossentry}{user5}{%
1374   \def\@glo@userv{#1}%
1375 }
```

user6

```
1376 \define@key{glossentry}{user6}{%
1377   \def\@glo@uservi{#1}%
1378 }
```

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.

```
1379 \define@key{glossentry}{short}{%
1380   \def\@glo@short{#1}%
1381 }
```

shortplural This key is provided for use by \newacronym.

```
1382 \define@key{glossentry}{shortplural}{%
1383   \def\@glo@shortpl{#1}%
1384 }
```

long This key is provided for use by \newacronym.

```
1385 \define@key{glossentry}{long}{%
1386   \def\@glo@long{#1}%
1387 }
```

longplural This key is provided for use by \newacronym.

```
1388 \define@key{glossentry}{longplural}{%
1389   \def\@glo@longpl{#1}%
1390 }
```

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

```
1391 \newcommand*\@glsnname{%
1392   \PackageError{glossaries}{name key required in
1393     \string\newglossaryentry\space for entry '\@glo@label'}{You
1394     haven't specified the entry name}}
```

\@glsnodels Define command to generate error if description key is missing.

```
1395 \newcommand*\@glsnodels{%
1396   \PackageError{glossaries}
1397   {%
1398     description key required in \string\newglossaryentry\space
1399     for entry '\@glo@label'%
1400   }%
1401   {%
1402     You haven't specified the entry description%
1403   }%
1404 }
```

```

\@glsdefaultplural  Now obsolete. Don't use.
1405 \newcommand*{\@glsdefaultplural}{}

s@missingnumberlist  Define a command to generate warning when numberlist not set.
1406 \newcommand*{\@gls@missingnumberlist}[1]{%
1407   ??%
1408   \ifglssavenumberlist
1409     \GlossariesWarning{Missing number list for entry ‘#1’.
1410       Maybe makeglossaries + rerun required.}%
1411   \else
1412     \PackageError{glossaries}%
1413     {Package option ‘savenumberlist=true’ required.}%
1414     {%
1415       You must use the ‘savenumberlist’ package option
1416       to reference location lists.%
1417     }%
1418   \fi
1419 }

\@glsdefaultsort  Define command to set default sort.
1420 \newcommand*{\@glsdefaultsort}{\@glo@name}

\gls@level  Register to increment entry levels.
1421 \newcount\gls@level

@gls@noexpand@field
1422 \newcommand{\@gls@noexpand@field}[3]{%
1423   \expandafter\global\expandafter
1424     \let\csname glo@#1@#2\endcsname#3%
1425 }

gls@noexpand@fields
1426 \newcommand{\@gls@noexpand@fields}[4]{%
1427   \ifcsdef{gls@assign@#3@field}
1428     {%
1429       \ifdefequal{#4}{\@gls@default@value}%
1430       {%
1431         \edef\@gls@value{\expandonce{#1}}%
1432         \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1433       }%
1434     }%
1435     \csuse{gls@assign@#3@field}{#2}{#4}%
1436   }%
1437 }%
1438 {%
1439   \ifdefequal{#4}{\@gls@default@value}%
1440   {%
1441     \edef\@gls@value{\expandonce{#1}}%

```

```

1442      \@gls@noexpand@field{#2}{#3}{\@gls@value}%
1443    }%
1444    {%
1445      \@gls@noexpand@field{#2}{#3}{#4}%
1446    }%
1447  }%
1448 }

```

\@gls@expand@field

```

1449 \newcommand{\@gls@expand@field}[3]{%
1450   \expandafter
1451   \protected@xdef\csname glo@#1@#2\endcsname{#3}%
1452 }

```

@gls@expand@fields

```

1453 \newcommand{\@gls@expand@fields}[4]{%
1454   \ifcsdef{gls@assign@#3@field}
1455   {%
1456     \ifdefequal{#4}{\@gls@default@value}%
1457     {%
1458       \edef\@gls@value{\expandonce{#1}}%
1459       \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1460     }%
1461     {%
1462       \expandafter\@gls@startswitexpandonce#4\relax\relax@gls@endcheck
1463     }%
1464     \@gls@expand@field{#2}{#3}{#4}%
1465   }%
1466   {%
1467     \csuse{gls@assign@#3@field}{#2}{#4}%
1468   }%
1469 }%
1470 }%
1471 {%
1472   \ifdefequal{#4}{\@gls@default@value}%
1473   {%
1474     \@gls@expand@field{#2}{#3}{#1}%
1475   }%
1476   {%
1477     \@gls@expand@field{#2}{#3}{#4}%
1478   }%
1479 }%
1480 }

```

tartswitexpandonce

```

1481 \def\@gls@expandonce{\expandonce}
1482 \def\@gls@startswitexpandonce#1#2@gls@endcheck#3#4{%
1483   \def\@gls@tmp{#1}%
1484   \ifdefequal{\@gls@expandonce}{\@gls@tmp}{#3}{#4}%

```

1485 }

`\gls@assign@field` `\gls@assign@field{<def value>}{<glossary type>}{<field>}{<tmp cs>}`

Assigns an entry field. Expansion performed by default (except for name, symbol and description where backward compatibility required). If `<tmp cs>` is `<@gls@default@value>`, `<def value>` is used instead.

1486 `\let\gls@assign@field\@gls@expand@fields`

`\glsexpandfields` Fully expand values when assigning fields (except for specific fields that are overridden by `\glssetnoexpandfield`).

1487 `\newcommand*{\glsexpandfields}{%`

1488 `\let\gls@assign@field\@gls@expand@fields`

1489 }

`\glsnoexpandfields` Don't expand values when assigning fields (except for specific fields that are overridden by `\glssetexpandfield`).

1490 `\newcommand*{\glsnoexpandfields}{%`

1491 `\let\gls@assign@field\@gls@noexpand@fields`

1492 }

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

1493 `\newrobustcmd{\newglossaryentry}[2]{%`

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

1494 `\glsdoifnoexists{#1}%`

1495 `{%`

1496 `\gls@defglossaryentry{#1}{#2}%`

1497 `}%`

1498 }

`\provideglossaryentry` Like `\newglossaryentry` but does nothing if the entry has already been defined.

1499 `\newrobustcmd{\provideglossaryentry}[2]{%`

1500 `\ifglstryexists{#1}%`

1501 `{}%`

1502 `{%`

1503 `\gls@defglossaryentry{#1}{#2}%`

1504 `}%`

1505 }

1506 `\@onlypreamble{\provideglossaryentry}`

`\new@glossaryentry` For use in document environment.

1507 `\newrobustcmd{\new@glossaryentry}[2]{%`

1508 `\ifundef\@gls@deffile`


```

1509 {%
1510     \global\newwrite\@gls@deffile
1511     \immediate\openout\@gls@deffile=\jobname.glsdefs
1512 }%
1513 {}%
1514 \ifglsentryexists{#1}{}%
1515 {%
1516     \gls@defglossaryentry{#1}{#2}%
1517 }%
1518 \@gls@writedef{#1}%
1519 }
1520 \AtBeginDocument
1521 {
1522     \makeatletter
1523     \InputIfFileExists{\jobname.glsdefs}{}{}%
1524     \makeatother
1525     \let\newglossaryentry\new@glossaryentry
1526 }
1527 \AtEndDocument{\ifdef\@gls@deffile{\closeout\@gls@deffile}{}%
1528 %     \end{macrocode}
1529 %\end{macro}
1530 %
1531 %\begin{macro}{\@gls@writedef}
1532 %\changes{3.10a}{2013-10-13}{new}
1533 % Writes glossary entry definition to \cs{@gls@deffile}.
1534 %     \begin{macrocode}
1535 \newcommand*{\@gls@writedef}[1]{%
1536     \immediate\write\@gls@deffile
1537     {%
1538         \string\ifglsentryexists{#1}{}\expandafter\@gobble\string\%^~J%
1539         \expandafter\@gobble\string\{\expandafter\@gobble\string\%^~J%
1540         \string\gls@defglossaryentry{#1}\expandafter\@gobble\string\%^~J%
1541         \expandafter\@gobble\string\{\expandafter\@gobble\string\%%
1542     }%

```

Write key value information:

```

1543 \@for\@gls@map:=\@gls@keymap\do
1544 {%
1545     \edef\glo@value{\expandafter\expandonce
1546         \csname glo@#1\expandafter\@secondoftwo\@gls@map\endcsname}%
1547     \@onelevel@sanitize\glo@value
1548     \immediate\write\@gls@deffile
1549     {%
1550         \expandafter\@firstoftwo\@gls@map
1551         =\expandafter\@gobble\string\{\glo@value\expandafter\@gobble\string\},%
1552         \expandafter\@gobble\string\%%
1553     }%
1554 }%

```

Provide hook:

```

1555 \glswritedefhook
1556 \immediate\write\@gls@deffile
1557 {%
1558     \expandafter\@gobble\string\%^~J%
1559     \expandafter\@gobble\string\}\expandafter\@gobble\string\%^~J%
1560     \expandafter\@gobble\string\}\expandafter\@gobble\string\%%
1561 }%
1562 }

```

`\@gls@keymap` List of entry definition key names and corresponding tag in control sequence used to store the value.

```

1563 \newcommand*{\@gls@keymap}{%
1564     {name}{name},%
1565     {sort}{sortvalue},% unescaped sort value
1566     {type}{type},%
1567     {first}{first},%
1568     {firstplural}{firstpl},%
1569     {text}{text},%
1570     {plural}{plural},%
1571     {description}{desc},%
1572     {descriptionplural}{descplural},%
1573     {symbol}{symbol},%
1574     {symbolplural}{symbolplural},%
1575     {user1}{useri},%
1576     {user2}{userii},%
1577     {user3}{useriii},%
1578     {user4}{useriv},%
1579     {user5}{userv},%
1580     {user6}{uservi},%
1581     {long}{long},%
1582     {longplural}{longpl},%
1583     {short}{short},%
1584     {shortplural}{shortpl},%
1585     {counter}{counter},%
1586     {parent}{parent}}%
1587 }

```

`\glsaddkey` `\glsaddkey{<key>}{<default value>}{<no link cs>}{<no link ucfirst cs>}{<link cs>}{<link ucfirst cs>}{<link allcaps cs>}`

Allow user to add their own custom keys.

```
1588 \newcommand*{\glsaddkey}{\@ifstar\@sglsaddkey\@glsaddkey}
```

Starred version switches on expansion for this key.

```

1589 \newcommand*{\@sglsaddkey}[1]{%
1590     \key@ifundefined{glossentry}{#1}%
1591     {%
1592         \expandafter\newcommand\expandafter*\expandafter

```

```

1593     {\csname gls@assign@#1@field\endcsname}[2]{%
1594       \@@gls@expand@field{##1}{#1}{##2}%
1595     }%
1596 }%
1597 {}%
1598 \@gls@saddkey{#1}%
1599 }

```

Unstarred version doesn't override default expansion.

```

1600 \newcommand*\@gls@saddkey[7]{%

```

Check the specified key doesn't already exist.

```

1601 \key@ifundefined{glossentry}{#1}%
1602 {%

```

Set up the key.

```

1603 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1604 \appto\@gls@keymap{,{#1}{#1}}%

```

Set the default value.

```

1605 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%

```

Assignment code.

```

1606 \appto\@newglossaryentryposthook{%
1607   \letcs{\@glo@tmp}{@glo@#1}%
1608   \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1609 }%

```

Define the no-link commands.

```

1610 \newcommand*{#3}[1]{\csuse{glo@##1@#1}}%
1611 \newcommand*{#4}[1]{%
1612   \letcs{\@glo@text}{glo@##1@#1}%
1613   \xmakefirstuc\@glo@text
1614 }%

```

Now for the commands with links. First the version with no case change:

```

1615 \ifcsdef{@gls@user@#1@}%
1616 {%
1617   \PackageError{glossaries}%
1618     {Can't define '\string#5' as helper command
1619     '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1620   }%
1621 }%
1622 {%
1623   \newrobustcmd*{#5}{\@ifstar{\csuse{@sgls@user@#1}}{\csuse{@gls@user@#1}}}%
1624   \expandafter\newcommand\expandafter*\expandafter
1625     {\csname @sgls@user@#1\endcsname}[1][1]{%
1626     \csuse{@gls@user@#1}[hyper=false,##1]%
1627   }%
1628   \expandafter\newcommand\expandafter*\expandafter
1629     {\csname @gls@user@#1\endcsname}[2][1]{%
1630     \new@ifnextchar[%

```

```

1631         {\csuse{@gls@user@#1@}{##1}{##2}}%
1632         {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
1633     \csdef{@gls@user@#1@}##1##2[##3]{%
1634         \glsdoifexists{##2}%
1635         {%
1636             \edef\@glo@type{\glsentrytype{##2}}%
1637             \@gls@link[##1]{##2}{#3{##2}##3}%
1638         }%
1639     }%
1640 }%

```

Next the version with the first letter converted to upper case:

```

1641     \ifcsdef{@Gls@user@#1@}%
1642     {%
1643         \PackageError{glossaries}%
1644         {Can't define '\string#6' as helper command
1645         '\expandafter\string\csname @Gls@user@#1@\endcsname' already exists}%
1646     }%
1647 }%
1648 {%
1649     \newrobustcmd*{#6}{\@ifstar{\csuse{@sGls@user@#1}}{\csuse{@Gls@user@#1}}}%
1650     \expandafter\newcommand\expandafter*\expandafter
1651     {\csname @sGls@user@#1\endcsname}[1][]{%
1652         \csuse{@Gls@user@#1}[hyper=false,##1]%
1653     }%
1654     \expandafter\newcommand\expandafter*\expandafter
1655     {\csname @Gls@user@#1\endcsname}[2][]{%
1656         \new@ifnextchar[%
1657             {\csuse{@Gls@user@#1@}{##1}{##2}}%
1658             {\csuse{@Gls@user@#1@}{##1}{##2}[]}}%
1659     \csdef{@Gls@user@#1@}##1##2[##3]{%
1660         \glsdoifexists{##2}%
1661         {%
1662             \edef\@glo@type{\glsentrytype{##2}}%
1663             \@gls@link[##1]{##2}{#4{##2}##3}%
1664         }%
1665     }%
1666 }%

```

Finally the all caps version:

```

1667     \ifcsdef{@GLS@user@#1@}%
1668     {%
1669         \PackageError{glossaries}%
1670         {Can't define '\string#7' as helper command
1671         '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1672     }%
1673 }%
1674 {%
1675     \newrobustcmd*{#7}{\@ifstar{\csuse{@sGLS@user@#1}}{\csuse{@GLS@user@#1}}}%
1676     \expandafter\newcommand\expandafter*\expandafter

```

```

1677         {\csname @sGLS@user@#1\endcsname}[1][]{%
1678         \csuse{@GLS@user@#1}[hyper=false,##1]%
1679         }%
1680     \expandafter\newcommand\expandafter*\expandafter
1681     {\csname @GLS@user@#1\endcsname}[2][]{%
1682     \new@ifnextchar[%
1683     {\csuse{@GLS@user@#1@}{##1}{##2}}%
1684     {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%
1685     \csdef{@GLS@user@#1@}##1##2[##3]{%
1686     \glsdoifexists{##2}%
1687     {%
1688     \edef\@glo@type{\glsentrytype{##2}}%
1689     \@gls@link[##1]{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1690     }%
1691     }%
1692 }%
1693 }%
1694 {%
1695     \PackageError{glossaries}{Key ‘#1’ already exists}{}%
1696 }%
1697 }

```

\gls.writedefhook

```

1698 \newcommand*\gls.writedefhook{}

```

\gls@assign@desc

```

1699 \newcommand*\gls@assign@desc[1]{%
1700     \gls@assign@field{#1}{desc}{\@glo@desc}%
1701     \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1702 }

```

ongnewglossaryentry

```

1703 \newcommand{\longnewglossaryentry}[3]{%
1704     \glsdoifnoexists{#1}%
1705     {%
1706         \bgroup
1707         \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1708         \long\def\@newglossaryentryprehook{%
1709             \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1710             \@org@newglossaryentryprehook
1711         }%
1712         \renewcommand*\gls@assign@desc[1]{%
1713             \global\cslet{glo@#1@desc}{\@glo@desc}%
1714             \global\cslet{glo@#1@descplural}{\@glo@desc}%
1715         }
1716         \gls@defglossaryentry{#1}{#2}%
1717     \egroup
1718 }
1719 }

```

Only allowed in the preamble. (Otherwise a long description could cause problems when writing the entry definition to the temporary file.)

```
1720 \@onlypreamble{\longnewglossaryentry}
```

`\provideglossaryentry` As the above but only defines the entry if it doesn't already exist.

```
1721 \newcommand{\longprovideglossaryentry}[3]{%
1722   \ifglentryexists{#1}{}%
1723   {\longnewglossaryentry{#1}{#2}{#3}}%
1724 }
1725 \@onlypreamble{\longprovideglossaryentry}
```

`\gls@defglossaryentry` `\gls@defglossaryentry{<label>}{<key-val list>}`

Defines a new entry without checking if it already exists.

```
1726 \newcommand{\gls@defglossaryentry}[2]{%
```

Store label

```
1727   \def\@glo@label{#1}%
```

Provide a means for user define keys to reference the label:

```
1728   \let\glslabel\@glo@label
```

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

```
1729   \let\@glo@name\@gls@name
```

```
1730   \let\@glo@desc\@gls@desc
```

```
1731   \let\@glo@descplural\@gls@default@value
```

```
1732   \let\@glo@type\@gls@default@value
```

```
1733   \let\@glo@symbol\@gls@default@value
```

```
1734   \let\@glo@symbolplural\@gls@default@value
```

```
1735   \let\@glo@text\@gls@default@value
```

```
1736   \let\@glo@plural\@gls@default@value
```

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

```
1737   \let\@glo@first\@gls@default@value
```

```
1738   \let\@glo@firstplural\@gls@default@value
```

Set the default sort:

```
1739   \let\@glo@sort\@gls@default@value
```

Set the default counter:

```
1740   \let\@glo@counter\@gls@default@value
```

```
1741   \def\@glo@see{}%
```

```

1742 \def\@glo@parent{}%
1743 \def\@glo@prefix{}%

1744 \def\@glo@useri{}%
1745 \def\@glo@userii{}%
1746 \def\@glo@useriii{}%
1747 \def\@glo@useriv{}%
1748 \def\@glo@userv{}%
1749 \def\@glo@uservi{}%

1750 \def\@glo@short{}%
1751 \def\@glo@shortpl{}%
1752 \def\@glo@long{}%
1753 \def\@glo@longpl{}%

```

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

```
1754 \@newglossaryentryprehook
```

Extract key-val information from third parameter:

```
1755 \setkeys{glossentry}{#2}%
```

Check there is a default glossary.

```

1756 \ifundef\glsdefaulttype
1757 {%
1758   \PackageError{glossaries}%
1759   {No default glossary type (have you used ‘nomain’?)}%
1760   {If you use package option ‘nomain’ you must define
1761    a new glossary before you can define entries}%
1762 }%
1763 {}%

```

Assign type. This must be fully expandable

```

1764 \gls@assign@field{\glsdefaulttype}{#1}{type}{\@glo@type}%
1765 \edef\@glo@type{\glsentrytype{#1}}%

```

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

```

1766 \ifcsundef{glolist@\@glo@type}%
1767 {%
1768   \PackageError{glossaries}%
1769   {Glossary type ‘\@glo@type’ has not been defined}%
1770   {You need to define a new glossary type, before making entries
1771    in it}%
1772 }%
1773 {%
1774   \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
1775   \expandafter\xdef\csname glolist@\@glo@type\endcsname{\@glolist@{#1},}%
1776 }%

```

Initialise level to 0.

```
1777 \gls@level=0\relax
```

Has this entry been assigned a parent?

```

1778   \ifx\@glo@parent\@empty

Doesn't have a parent. Set \glo@<label>@parent to empty.
1779   \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1780   \else

Has a parent. Check to ensure this entry isn't its own parent.
1781   \ifthenelse{\equal{#1}{\@glo@parent}}{%
1782   {%
1783   \PackageError{glossaries}{Entry '#1' can't be its own parent}{}%
1784   \def\@glo@parent{}%
1785   \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1786   }%
1787   {%

Check the parent exists:
1788   \ifglentryexists{\@glo@parent}%
1789   {%

Parent exists. Set \glo@<label>@parent.
1790   \expandafter\xdef\csname glo@#1@parent\endcsname{\@glo@parent}%

Determine level.
1791   \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1792   \advance\gls@level by 1\relax

If name hasn't been specified, use same as the parent name
1793   \ifx\@glo@name\@glsnoname
1794   \expandafter\let\expandafter\@glo@name
1795   \csname glo@\@glo@parent @name\endcsname

If name and plural haven't been specified, use same as the parent
1796   \ifx\@glo@plural\@gls@default@value
1797   \expandafter\let\expandafter\@glo@plural
1798   \csname glo@\@glo@parent @plural\endcsname
1799   \fi
1800   \fi
1801   }%
1802   {%

Parent doesn't exist, so issue an error message and change this entry to have no
parent
1803   \PackageError{glossaries}%
1804   {%
1805   Invalid parent '\@glo@parent'
1806   for entry '#1' - parent doesn't exist%
1807   }%
1808   {%
1809   Parent entries must be defined before their children%
1810   }%
1811   \def\@glo@parent{}%

```



```

1812         \expandafter\gdef\csname glo@#1@parent\endcsname{%
1813     }%
1814 }%
1815 \fi

Set the level for this entry
1816 \expandafter\xdef\csname glo@#1@level\endcsname{\number\gls@level}%

Define commands associated with this entry:
1817 \gls@assign@field{\@glo@name}{#1}{sortvalue}{\@glo@sort}%
1818 \letcs\@glo@sort{glo@#1@sortvalue}%
1819 \gls@assign@field{\@glo@name}{#1}{text}{\@glo@text}%
1820 \expandafter\gls@assign@field\expandafter
1821     {\csname glo@#1@text\endcsname\glspluralsuffix}%
1822     {#1}{plural}{\@glo@plural}%
1823 \expandafter\gls@assign@field\expandafter
1824     {\csname glo@#1@text\endcsname}%
1825     {#1}{first}{\@glo@first}%

If first has been specified, make the default by appending \glspluralsuffix,
otherwise make the default the value of the plural key.
1826 \ifx\@glo@first\@gls@default@value
1827     \expandafter\gls@assign@field\expandafter
1828         {\csname glo@#1@plural\endcsname}%
1829         {#1}{firstpl}{\@glo@firstplural}%
1830 \else
1831     \expandafter\gls@assign@field\expandafter
1832         {\csname glo@#1@first\endcsname\glspluralsuffix}%
1833         {#1}{firstpl}{\@glo@firstplural}%
1834 \fi

1835 \ifcsundef{@glo@type@\@glo@type @counter}%
1836 {%
1837     \def\@glo@defaultcounter{\glscounter}%
1838 }%
1839 {%
1840     \letcs\@glo@defaultcounter{@glo@type@\@glo@type @counter}%
1841 }%
1842 \gls@assign@field{\@glo@defaultcounter}{#1}{counter}{\@glo@counter}%
1843 \gls@assign@field{}{#1}{useri}{\@glo@useri}%
1844 \gls@assign@field{}{#1}{userii}{\@glo@userii}%
1845 \gls@assign@field{}{#1}{useriii}{\@glo@useriii}%
1846 \gls@assign@field{}{#1}{useriv}{\@glo@useriv}%
1847 \gls@assign@field{}{#1}{userv}{\@glo@userv}%
1848 \gls@assign@field{}{#1}{uservi}{\@glo@uservi}%
1849 \gls@assign@field{}{#1}{short}{\@glo@short}%
1850 \gls@assign@field{}{#1}{shortpl}{\@glo@shortpl}%
1851 \gls@assign@field{}{#1}{long}{\@glo@long}%
1852 \gls@assign@field{}{#1}{longpl}{\@glo@longpl}%
1853 \ifx\@glo@name\@gls@name
1854     \@gls@name

```

```

1855     \let\@glo@name\@gls@default@value
1856     \fi
1857     \gls@assign@field{#1}{name}{\@glo@name}%

```

Set default numberlist if not defined:

```

1858     \ifcsundef{glo@#1@numberlist}%
1859     {%
1860         \csxdef{glo@#1@numberlist}{\noexpand\@gls@missingnumberlist{\@glo@label}}%
1861     }%
1862     {}%

```

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.

```

1863     \def\@glo@@desc{\@glo@first}%
1864     \ifx\@glo@desc\@glo@@desc
1865         \let\@glo@desc\@glo@first
1866     \fi
1867     \ifx\@glo@desc\@gls@nodesc
1868         \@gls@nodesc
1869         \let\@glo@desc\@gls@default@value
1870     \fi
1871     \gls@assign@desc{#1}%

```

Set the sort key for this entry:

```

1872     \@gls@defsort{\@glo@type}{#1}%

1873     \def\@glo@@symbol{\@glo@text}%
1874     \ifx\@glo@symbol\@glo@@symbol
1875         \let\@glo@symbol\@glo@text
1876     \fi
1877     \gls@assign@field{\relax}{#1}{symbol}{\@glo@symbol}%
1878     \expandafter
1879         \gls@assign@field\expandafter
1880         {\csname glo@#1@symbol\endcsname}
1881         {#1}{symbolplural}{\@glo@symbolplural}%

```

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

```

1882     \expandafter\gdef\csname glo@#1@flagfalse\endcsname{%
1883         \expandafter\global\expandafter
1884             \let\csname ifglo@#1@flag\endcsname\iffalse
1885     }%
1886     \expandafter\gdef\csname glo@#1@flagtrue\endcsname{%
1887         \expandafter\global\expandafter
1888             \let\csname ifglo@#1@flag\endcsname\iftrue
1889     }%
1890     \csname glo@#1@flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

1891     \ifx\@glo@see\@empty

```

```

1892 \else
1893 \protected@edef\@do@glsssee{%
1894 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
1895 \noexpand\@nil
1896 \noexpand\expandafter\noexpand\@glsssee\noexpand\@glo@list{#1}}%
1897 \@do@glsssee
1898 \fi

```

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

```
1899 \do@glo@storeentry{#1}%
```

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

```

1900 \@newglossaryentryposthook
1901 }

```

glossaryentryprehook Allow extra information to be added to glossary entries:

```
1902 \newcommand*{\@newglossaryentryprehook}{}

```

glossaryentryposthook Allow extra information to be added to glossary entries:

```
1903 \newcommand*{\@newglossaryentryposthook}{}

```

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

```

1904 \newcommand*{\glsmoveentry}[2]{%
1905 \edef\glo@type{\csname glo@#1@type\endcsname}%
1906 \def\glo@list{,%}%
1907 \for@gl@entries[\glo@type]{\glo@label}%
1908 {%
1909 \ifthenelse{\equal{\glo@label}{#1}}{\eappto\glo@list{\glo@label,}}%
1910 }%
1911 \cslet{glolist@\glo@type}{\glo@list}%
1912 \csdef{glo@#1@type}{#2}%
1913 }

```

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

```

1914 \ifgl@xindy
1915 \newcommand*{\@glossaryentryfield}{\string\glossentry}
1916 \else
1917 \newcommand*{\@glossaryentryfield}{\string\glossentry}
1918 \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.)

```

1919 \ifgl@xindy
1920 \newcommand*{\@glossarysubentryfield}{%

```

```

1921 \string\subglossentry}
1922 \else
1923 \newcommand*{\@glossarysubentryfield}{%
1924 \string\subglossentry}
1925 \fi

```

\@glo@storeentry \@glo@storeentry{<label>}

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.)

```

1926 \newcommand{\@glo@storeentry}[1]{%

```

Escape special characters in the label:

```

1927 \def\@glo@label{#1}%
1928 \@gls@checkmkidxchars\@glo@label

```

Get the sort string and escape any special characters

```

1929 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
1930 \@gls@checkmkidxchars\@glo@sort

```

Same again for the name string. Escape any special characters in the prefix

```

1931 \@gls@checkmkidxchars\@glo@prefix

```

Get the parent, if one exists

```

1932 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%

```

Write the information to the glossary file.

```

1933 \ifglxindy

```

Store using xindy syntax.

```

1934 \ifx\@glo@parent\@empty

```

Entry doesn't have a parent

```

1935 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1936 (\string\@glo@sort\string" %
1937 \string\@glo@prefix\@glossarysubentryfield{\@glo@label}\string") %
1938 }%
1939 \else

```

Entry has a parent

```

1940 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1941 \csname glo@\@glo@parent @index\endcsname
1942 (\string\@glo@sort\string" %
1943 \string\@glo@prefix\@glossarysubentryfield
1944 {\csname glo@#1@level\endcsname}{\@glo@label}\string") %
1945 }%
1946 \fi
1947 \else

```

```

Store using makeindex syntax.
1948 \ifx\@glo@parent\@empty
Sanitize \@glo@prefix
1949 \@onelevel@sanitize\@glo@prefix
Entry doesn't have a parent
1950 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1951 \@glo@sort\@gls@actualchar\@glo@prefix
1952 \@glossaryentryfield{\@glo@label}%
1953 }%
1954 \else
Entry has a parent
1955 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1956 \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
1957 \@glo@sort\@gls@actualchar\@glo@prefix
1958 \@glossarysubentryfield
1959 {\csname glo@#1@level\endcsname}{\@glo@label}%
1960 }%
1961 \fi
1962 \fi
1963 }

```

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, but first we need to know if we're in `amsmath`'s align environment's measuring pass.

```

\gls@ifnotmeasuring
1964 \AtBeginDocument{%
1965 \@ifpackageloaded{amsmath}%
1966 {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
1967 }%
1968 }
1969 \newcommand*\@gls@ifnotmeasuring[1]{%
1970 \ifmeasuring@
1971 \else
1972 #1%
1973 \fi
1974 }
1975 \newcommand*\gls@ifnotmeasuring[1]{#1}

```

`\glsreset` 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.

```

1976 \newcommand*\glsreset[1]{%

```

```

1977 \gls@ifnotmeasuring
1978 {%
1979 \glsdoifexists{#1}%
1980 {%
1981 \expandafter\global\csname glo@#1@flagfalse\endcsname
1982 }%
1983 }%
1984 }

```

`\glslocalreset` As above, but with only a local effect:

```

1985 \newcommand*{\glslocalreset}[1]{%
1986 \gls@ifnotmeasuring
1987 {%
1988 \glsdoifexists{#1}%
1989 {%
1990 \expandafter\let\csname ifglo@#1@flag\endcsname\iffalse
1991 }%
1992 }%
1993 }

```

`\glsunset` 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.

```

1994 \newcommand*{\glsunset}[1]{%
1995 \gls@ifnotmeasuring
1996 {%
1997 \glsdoifexists{#1}%
1998 {%
1999 \expandafter\global\csname glo@#1@flagtrue\endcsname
2000 }%
2001 }%
2002 }

```

`\glslocalunset` As above, but with only a local effect:

```

2003 \newcommand*{\glslocalunset}[1]{%
2004 \gls@ifnotmeasuring
2005 {%
2006 \glsdoifexists{#1}%
2007 {%
2008 \expandafter\let\csname ifglo@#1@flag\endcsname\iftrue
2009 }%
2010 }%
2011 }

```

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

Syntax: `\glsresetall[<glossary-list>]`

`\glsresetall`

```

2012 \newcommand*{\glsresetall}[1][\@glo@types]{%
2013 \forallglsentries[#1]{\@glsentry}%

```

```

2014  {%
2015    \glsreset{\@glsentry}%
2016  }%
2017 }

```

As above, but with only a local effect:

`\glslocalresetall`

```

2018 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
2019   \forallglsentries[#1]{\@glsentry}%
2020   {%
2021     \glslocalreset{\@glsentry}%
2022   }%
2023 }

```

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

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

`\glsunsetall`

```

2024 \newcommand*{\glsunsetall}[1][\@glo@types]{%
2025   \forallglsentries[#1]{\@glsentry}%
2026   {%
2027     \glsunset{\@glsentry}%
2028   }%
2029 }

```

As above, but with only a local effect:

`\glslocalunsetall`

```

2030 \newcommand*{\glslocalunsetall}[1][\@glo@types]{%
2031   \forallglsentries[#1]{\@glsentry}%
2032   {%
2033     \glslocalunset{\@glsentry}%
2034   }%
2035 }

```

1.9 Loading files containing glossary entries

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

`\loadglsentries[⟨type⟩]{⟨filename⟩}`

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

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

`\loadglsentries`

```
2036 \newcommand*{\loadglsentries}[2][\@gls@default]{%
2037   \let\@gls@default\glsdefaulttype
2038   \def\glsdefaulttype{#1}\input{#2}%
2039   \let\glsdefaulttype\@gls@default
2040 }
```

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

```
2041 \@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`

```
2042 \newcommand*{\glstextformat}[1]{#1}
```

`\glsentryfmt` As from version 3.11a, the way in which an entry is displayed is now governed by `\glsentryfmt`. This doesn’t take any arguments. The required information is set by commands like `\gls`. To ensure backward compatibility, the default use the old `\glsdisplay` and `\glsdisplayfirst` style of commands

```
2043 \newcommand*{\glsentryfmt}{%
2044   \@@gls@default@entryfmt\glsdisplayfirst\glsdisplay
2045 }
```

Format that provides backwards compatibility:

```
2046 \newcommand*{\@@gls@default@entryfmt}[2]{%
2047   \ifdefempty\glscustomtext
2048     {%
2049       \glsifplural
2050       {%
```

Plural form

```
2051       \glscapscase
2052       {%
```

Don’t adjust case

```
2053       \ifglsused\glslabel
2054       {%
```

Subsequent use

```
2055       #2{\glsentryplural{\glslabel}}%
2056       {\glsentrydescplural{\glslabel}}%
```



```

2057         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2058     }%
2059     {%

```

First use

```

2060         #1{\glsentryfirstplural{\glslabel}}%
2061         {\glsentrydescplural{\glslabel}}%
2062         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2063     }%
2064 }%
2065 {%

```

Make first letter upper case

```

2066     \ifglsused\glslabel
2067     {%

```

Subsequent use. (Expansion was used in version 3.07 and below in case the name wasn't the first thing to be displayed, but now the user can sort out the upper casing in `\defglsentryfmt`, which avoids the issues caused by fragile commands.)

```

2068     \ifbool{glscompatible-3.07}%
2069     {%
2070         \protected@edef\@glo@etext{%
2071             #2{\glsentryplural{\glslabel}}%
2072             {\glsentrydescplural{\glslabel}}%
2073             {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2074         \xmakefirstuc\@glo@etext
2075     }%
2076     {%
2077         #2{\Glsentryplural{\glslabel}}%
2078         {\glsentrydescplural{\glslabel}}%
2079         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2080     }%
2081 }%
2082 {%

```

First use

```

2083     \ifbool{glscompatible-3.07}%
2084     {%
2085         \protected@edef\@glo@etext{%
2086             #1{\glsentryfirstplural{\glslabel}}%
2087             {\glsentrydescplural{\glslabel}}%
2088             {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2089         \xmakefirstuc\@glo@etext
2090     }%
2091     {%
2092         #1{\Glsentryfirstplural{\glslabel}}%
2093         {\glsentrydescplural{\glslabel}}%
2094         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2095     }%
2096 }%

```

2097 }%
 2098 {%

Make all upper case

2099 \ifglsused\glslabel
 2100 {%

Subsequent use

2101 \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}}%
 2102 {\glsentrydescplural{\glslabel}}}%
 2103 {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
 2104 }%
 2105 {%

First use

2106 \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}}%
 2107 {\glsentrydescplural{\glslabel}}}%
 2108 {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
 2109 }%
 2110 }%
 2111 }%
 2112 {%

Singular form

2113 \glscapscase
 2114 {%

Don't adjust case

2115 \ifglsused\glslabel
 2116 {%

Subsequent use

2117 #2{\glsentrytext{\glslabel}}}%
 2118 {\glsentrydesc{\glslabel}}}%
 2119 {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
 2120 }%
 2121 {%

First use

2122 #1{\glsentryfirst{\glslabel}}}%
 2123 {\glsentrydesc{\glslabel}}}%
 2124 {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
 2125 }%
 2126 }%
 2127 {%

Make first letter upper case

2128 \ifglsused\glslabel
 2129 {%

Subsequent use

2130 \ifbool{glscompatible-3.07}%
 2131 {%

```

2132         \protected@edef\@glo@etext{%
2133             #2{\glsentrytext{\glslabel}}}%
2134             {\glsentrydesc{\glslabel}}}%
2135             {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2136         \xmakefirstuc\@glo@etext
2137     }%
2138     {%
2139         #2{\Glsentrytext{\glslabel}}}%
2140         {\glsentrydesc{\glslabel}}}%
2141         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2142     }%
2143 }%
2144 {%

```

First use

```

2145         \ifbool{glscompatible-3.07}%
2146         {%
2147             \protected@edef\@glo@etext{%
2148                 #1{\glsentryfirst{\glslabel}}}%
2149                 {\glsentrydesc{\glslabel}}}%
2150                 {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2151             \xmakefirstuc\@glo@etext
2152         }%
2153         {%
2154             #1{\Glsentryfirst{\glslabel}}}%
2155             {\glsentrydesc{\glslabel}}}%
2156             {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2157         }%
2158     }%
2159 }%
2160 {%

```

Make all upper case

```

2161         \ifglsused\glslabel
2162         {%

```

Subsequent use

```

2163         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}}%
2164         {\glsentrydesc{\glslabel}}}%
2165         {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2166     }%
2167     {%

```

First use

```

2168         \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}}%
2169         {\glsentrydesc{\glslabel}}}%
2170         {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2171     }%
2172 }%
2173 }%
2174 }%

```

```

2175  {%
      Custom text provided in \glsdisp
2176  \ifglsused{\glslabel}%
2177  {%
      Subsequent use
2178      #2{\glscustomtext}%
2179      {\glsentrydesc{\glslabel}}%
2180      {\glsentrysymbol{\glslabel}}{%}%
2181  }%
2182  {%
      First use
2183      #1{\glscustomtext}%
2184      {\glsentrydesc{\glslabel}}%
2185      {\glsentrysymbol{\glslabel}}{%}%
2186  }%
2187  }%
2188 }

```

`\glsgenentryfmt` Define a generic format that just uses the first, text, plural or first plural keys (or the custom text) with the insert text appended.

```

2189 \newcommand*{\glsgenentryfmt}{%
2190   \ifdefempty\glscustomtext
2191   {%
2192     \glsifplural
2193     {%
      Plural form
2194       \glscapscase
2195       {%
        Don't adjust case
2196         \ifglsused\glslabel
2197         {%
          Subsequent use
2198           \glsentryplural{\glslabel}\glsinsert
2199           }%
2200           {%
            First use
2201             \glsentryfirstplural{\glslabel}\glsinsert
2202             }%
2203             }%
2204             {%
              Make first letter upper case
2205               \ifglsused\glslabel
2206               {%

```

Subsequent use.

```
2207      \Glsentryplural{\glslabel}\glsinsert
2208      }%
2209      {%
```

First use

```
2210      \Glsentryfirstplural{\glslabel}\glsinsert
2211      }%
2212      }%
2213      {%
```

Make all upper case

```
2214      \ifglsused\glslabel
2215      {%
```

Subsequent use

```
2216      \mfirstucMakeUppercase
2217      {\glsentryplural{\glslabel}\glsinsert}%
2218      }%
2219      {%
```

First use

```
2220      \mfirstucMakeUppercase
2221      {\glsentryfirstplural{\glslabel}\glsinsert}%
2222      }%
2223      }%
2224      }%
2225      {%
```

Singular form

```
2226      \glscapscase
2227      {%
```

Don't adjust case

```
2228      \ifglsused\glslabel
2229      {%
```

Subsequent use

```
2230      \glsentrytext{\glslabel}\glsinsert
2231      }%
2232      {%
```

First use

```
2233      \glsentryfirst{\glslabel}\glsinsert
2234      }%
2235      }%
2236      {%
```

Make first letter upper case

```
2237      \ifglsused\glslabel
2238      {%
```

Subsequent use

```
2239      \Glsentrytext{\glslabel}\glsinsert
2240      }%
2241      {%
```

First use

```
2242      \Glsentryfirst{\glslabel}\glsinsert
2243      }%
2244      }%
2245      {%
```

Make all upper case

```
2246      \ifglsused\glslabel
2247      {%
```

Subsequent use

```
2248      \mfirstucMakeUppercase{\glsentrytext{\glslabel}\glsinsert}%
2249      }%
2250      {%
```

First use

```
2251      \mfirstucMakeUppercase{\glsentryfirst{\glslabel}\glsinsert}%
2252      }%
2253      }%
2254      }%
2255      }%
2256      {%
```

Custom text provided in `\glsdisp`. (The insert is most likely to be empty at this point.)

```
2257      \glscustomtext\glsinsert
2258      }%
2259 }
```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2260 \newcommand*{\glsdisplayfirst}[4]{#1#4}
```

`\glsdisplay` Deprecated. Kept for backward compatibility.

```
2261 \newcommand*{\glsdisplay}[4]{#1#4}
```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2262 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2263   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2264   Use \string\defglsentryfmt\space instead}%
2265   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2266   \edef\@gls@doentrydef{%
2267     \noexpand\defglsentryfmt[#1]{%
2268       \noexpand\ifcsdef{gls@#1@displayfirst}%
2269       {%
2270         \noexpand\@@gls@default@entryfmt
```

```

2271         {\noexpand\csuse{gls@#1@displayfirst}}
2272         {\noexpand\csuse{gls@#1@display}}}%
2273     }%
2274     {%
2275         \noexpand\@@gls@default@entryfmt
2276         {\noexpand\glsdisplayfirst}
2277         {\noexpand\csuse{gls@#1@display}}}%
2278     }%
2279 }%
2280 }%
2281 \@gls@doentrydef
2282 }

```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```

2283 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
2284   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2285   Use \string\defglsentryfmt\space instead}%
2286   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2287   \edef\@gls@doentrydef{%
2288     \noexpand\defglsentryfmt[#1]{%
2289       \noexpand\ifcsdef{gls@#1@display}%
2290       {%
2291         \noexpand\@@gls@default@entryfmt
2292         {\noexpand\csuse{gls@#1@displayfirst}}
2293         {\noexpand\csuse{gls@#1@display}}}%
2294       }%
2295       {%
2296         \noexpand\@@gls@default@entryfmt
2297         {\noexpand\csuse{gls@#1@displayfirst}}}%
2298         {\noexpand\glsdisplay}%
2299       }%
2300     }%
2301   }%
2302   \@gls@doentrydef
2303 }

```

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` and `\glsdisp`, 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 `\defentryfmt`). 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.

```

2304 \define@key{glslink}{counter}{%
2305   \ifcsundef{c@#1}%
2306   {%
2307     \PackageError{glossaries}%
2308     {There is no counter called ‘#1’}%
2309     {%
2310       The counter key should have the name of a valid counter
2311       as its value%
2312     }%
2313   }%
2314   {%
2315     \def\@gls@counter{#1}%
2316   }%
2317 }
```

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.

```

2318 \define@key{glslink}{format}{%
2319 \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.

```

2320 \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.

```

2321 \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
2322 \newrobustcmd*{\glslink}{%
2323 \ifstar\@sgls@link\@gls@link
2324 }

\@sgls@link The starred version of \glslink calls the unstarred version with hyperlinks dis-
abled.
2325 \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.
2326 \newcommand*{\@gls@link}[3][\@gls@link[hyper=false,#1]]{%
2327 \ifglsentryexists{#2}%
2328 {%
2329 \@gls@link[#1]{#2}{#3}%
2330 }{%
2331 \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2332 defined}{You need to define a glossary entry before you
2333 can use it.}%

Display the specified text. (The entry doesn't exist so there's nothing to link it
to.)
2334 \glstextformat{#3}%
2335 }%
2336 }

\@gls@link
2337 \def\@gls@link[#1]#2#3{%

Inserting \leavevmode suggested by Donald Arseneau (avoids problem with
tabularx).
2338 \leavevmode
2339 \def\glslabel{#2}%
2340 \def\@glsnumberformat{\glsnumberformat}%
2341 \edef\@gls@counter{\csname glo@#2@counter\endcsname}%

If this is in one of the “nohypertypes” glossaries, suppress the hyperlink by de-
fault
2342 \edef\gls@type{\csname glo@#2@type\endcsname}%
2343 \expandafter\DTLifinlist\expandafter
2344 {\gls@type}{\@gls@nohyperlist}%
2345 {%
2346 \KV@glslink@hyperfalse
2347 }%
2348 {%
2349 \KV@glslink@hypertrue
2350 }%
2351 \setkeys{glslink}{#1}%

```

Store the entry's counter in \theglsentrycounter

```

2352 \gls@saveentrycounter
    Define sort key if necessary:
2353 \gls@setsort{#2}%

2354 \do@wrglossary{#2}%
2355 \ifKV@glslink@hyper
2356 \glslink{\glolinkprefix#2}{\glstextformat{#3}}%
2357 \else

2358 \glstextformat{#3}%
2359 \fi
2360 }
```

\glolinkprefix

```
2361 \newcommand*{\glolinkprefix}{glo:}
```

\glsentrycounter Set default value of entry counter

```
2362 \def\glsentrycounter{\glscounter}%
```

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

```
2363 \newcommand*{\gls@saveentrycounter}{%
2364 \def\gls@Hcounter{}}
```

Are we using equation counter?

```
2365 \ifthenelse{\equal{\gls@counter}{equation}}%
2366 {
```

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

```

2367 \ifcsundef{xatlevel@}%
2368 {%
2369 \edef\theglsentrycounter{\expandafter\noexpand
2370 \csname the\gls@counter\endcsname}%
2371 }%
2372 {%
2373 \ifx\xatlevel@\empty
2374 \edef\theglsentrycounter{\expandafter\noexpand
2375 \csname the\gls@counter\endcsname}%
2376 \else
2377 \savecounters@
2378 \advance\c@equation by 1\relax
2379 \edef\theglsentrycounter{\csname the\gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

2380 \ifcsundef{theH\gls@counter}%
2381 {%
2382 \def\gls@Hcounter{\theglsentrycounter}%
2383 }
```

```

2384      {%
2385      \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2386      }%
2387      \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2388      \restorecounters@
2389    \fi
2390  }%
2391 }%
2392 {%

```

Not using equation counter so no special measures:

```

2393    \edef\theglentrycounter{\expandafter\noexpand
2394    \csname the\@gls@counter\endcsname}%
2395  }%

```

Check if hyperref version of this counter

```

2396  \ifx\@gls@Hcounter\@empty
2397    \ifcsundef{theH\@gls@counter}%
2398    {%
2399      \def\theHglentrycounter{\theglentrycounter}%
2400    }%
2401    {%
2402      \protected@edef\theHglentrycounter{\expandafter\noexpand
2403      \csname theH\@gls@counter\endcsname}%
2404    }%
2405  \fi
2406 }

```

`\@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.

```

2407 \def\@set@glo@numformat#1#2#3#4{%
2408   \expandafter\@glo@check@mkidrangechar#3\@nil
2409   \protected@edef#1{%
2410     \@glo@prefix setentrycounter[#4]{#2}%
2411     \expandafter\string\csname\@glo@suffix\endcsname
2412   }%
2413   \@gls@checkmkidchars#1%
2414 }

```

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.

```

2415 \def\@glo@check@mkidrangechar#1#2\@nil{%
2416 \if#1(\relax

```

```

2417 \def\@glo@prefix{()%
2418 \if\relax#2\relax
2419 \def\@glo@suffix{glsnumberformat}%
2420 \else
2421 \def\@glo@suffix{#2}%
2422 \fi
2423 \else
2424 \if#1)\relax
2425 \def\@glo@prefix{}}%
2426 \if\relax#2\relax
2427 \def\@glo@suffix{glsnumberformat}%
2428 \else
2429 \def\@glo@suffix{#2}%
2430 \fi
2431 \else
2432 \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2433 \fi
2434 \fi}

```

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

```

2435 \newcommand*{\@gls@escbsdq}[1]{%
2436 \def\@gls@checkedmkidx{}%
2437 \let\gls@xdystring=#1\relax
2438 \@onelevel@sanitize\gls@xdystring
2439 \edef\do@gls@xdycheckbackslash{%
2440 \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2441 \@backslashchar\@backslashchar\noexpand\null}%
2442 \do@gls@xdycheckbackslash
2443 \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
2444 \def\@gls@checkedmkidx{}%
2445 \expandafter\@gls@xdycheckquote\gls@xdystring\@nil""\null
2446 \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%

```

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

```

2447 \@for\@gls@tmp:=\gls@protected@pagefmts\do
2448 {%
2449 \edef\@gls@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gls@tmp}%
2450 \@onelevel@sanitize\@gls@sanitized@tmp
2451 \edef\gls@dostubst{%
2452 \noexpand\DTLsubstituteall\noexpand\gls@xdystring
2453 {\@gls@sanitized@tmp}{\expandonce\@gls@tmp}%
2454 }%
2455 \gls@dostubst
2456 }%

```

Assign to required control sequence

```

2457 \let#1=\gls@xdystring
2458 }

```

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

`\gls@checkmkidxchars`

```

2459 \newcommand{\@gls@checkmkidxchars}[1]{%
2460   \ifglxsindy
2461     \@gls@escbsdq{#1}%
2462   \else
2463     \def\@gls@checkedmkidx{%
2464       \expandafter\@gls@checkquote#1\@nil""\null
2465       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2466     \def\@gls@checkedmkidx{%
2467       \expandafter\@gls@checkescquote#1\@nil"\\"\null
2468       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2469     \def\@gls@checkedmkidx{%
2470       \expandafter\@gls@checkescactual#1\@nil"??\null
2471       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2472     \def\@gls@checkedmkidx{%
2473       \expandafter\@gls@checkactual#1\@nil??\null
2474       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2475     \def\@gls@checkedmkidx{%
2476       \expandafter\@gls@checkbar#1\@nil||\null
2477       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2478     \def\@gls@checkedmkidx{%
2479       \expandafter\@gls@checkescbar#1\@nil\\|\null
2480       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2481     \def\@gls@checkedmkidx{%
2482       \expandafter\@gls@checklevel#1\@nil!!\null
2483       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2484     \fi
2485 }
```

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

`\@gls@updatechecked`

```

2486 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}
```

`\@gls@tmpb` Define temporary token

```

2487 \newtoks\@gls@tmpb
```

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

```

2488 \def\@gls@checkquote#1"#2"#3\@nil{%
2489   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2490   \toks@={#1}%
2491   \ifx\@nil#2\@nil
2492     \ifx\@nil#3\@nil
2493       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2494     \def\@gls@checkquote{\relax}%
2495   \else
2496     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2497       \@gls@quotechar\@gls@quotechar\@gls@quotechar}%

```

```

2498 \def\@gls@checkquote{\@gls@checkquote#3\null}%
2499 \fi
2500 \else
2501 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2502 \@gls@quotearchar\@gls@quotearchar}%
2503 \ifx\null#3\null
2504 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2505 \else
2506 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2507 \fi
2508 \fi
2509 \@gls@checkquote
2510 }

```

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

```

2511 \def\@gls@checkescquote#1\"#2\"#3\null{%
2512 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2513 \toks@={#1}%
2514 \ifx\null#2\null
2515 \ifx\null#3\null
2516 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2517 \def\@gls@checkescquote{\relax}%
2518 \else
2519 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2520 \@gls@quotearchar\string\" \@gls@quotearchar
2521 \@gls@quotearchar\string\" \@gls@quotearchar}%
2522 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2523 \fi
2524 \else
2525 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2526 \@gls@quotearchar\string\" \@gls@quotearchar}%
2527 \ifx\null#3\null
2528 \def\@gls@checkescquote{\@gls@checkescquote#2\"\" \null}%
2529 \else
2530 \def\@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
2531 \fi
2532 \fi
2533 \@gls@checkescquote
2534 }

```

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

```

2535 \def\@gls@checkescactual#1\?#2\?#3\null{%
2536 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2537 \toks@={#1}%
2538 \ifx\null#2\null
2539 \ifx\null#3\null
2540 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2541 \def\@gls@checkescactual{\relax}%
2542 \else

```

```

2543 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2544 \@gls@quotearchar\string\"@gls@actualchar
2545 \@gls@quotearchar\string\"@gls@actualchar}%
2546 \def\@@gls@checkescactual{\@gls@checkescactual#3\null}%
2547 \fi
2548 \else
2549 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2550 \@gls@quotearchar\string\"@gls@actualchar}%
2551 \ifx\null#3\null
2552 \def\@@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2553 \else
2554 \def\@@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2555 \fi
2556 \fi
2557 \@@gls@checkescactual
2558 }

```

\@gls@checkescbar Similarly for \|:

```

2559 \def\@gls@checkescbar#1\|#2\|#3\null{%
2560 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2561 \toks@={#1}%
2562 \ifx\null#2\null
2563 \ifx\null#3\null
2564 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2565 \def\@@gls@checkescbar{\relax}%
2566 \else
2567 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2568 \@gls@quotearchar\string\"@gls@encapchar
2569 \@gls@quotearchar\string\"@gls@encapchar}%
2570 \def\@@gls@checkescbar{\@gls@checkescbar#3\null}%
2571 \fi
2572 \else
2573 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2574 \@gls@quotearchar\string\"@gls@encapchar}%
2575 \ifx\null#3\null
2576 \def\@@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2577 \else
2578 \def\@@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2579 \fi
2580 \fi
2581 \@@gls@checkescbar
2582 }

```

\@gls@checkesclevel Similarly for \!:

```

2583 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2584 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2585 \toks@={#1}%
2586 \ifx\null#2\null
2587 \ifx\null#3\null

```

```

2588 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2589 \def\@@gls@checkesclevel{\relax}%
2590 \else
2591 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2592 \@gls@quotechar\string"\@gls@levelchar
2593 \@gls@quotechar\string"\@gls@levelchar}%
2594 \def\@@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2595 \fi
2596 \else
2597 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2598 \@gls@quotechar\string"\@gls@levelchar}%
2599 \ifx\null#3\null
2600 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2601 \else
2602 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2603 \fi
2604 \fi
2605 \@@gls@checkesclevel
2606 }

```

\@gls@checkbar and for |:

```

2607 \def\@gls@checkbar#1|#2|#3\null{%
2608 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2609 \toks@={#1}%
2610 \ifx\null#2\null
2611 \ifx\null#3\null
2612 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2613 \def\@@gls@checkbar{\relax}%
2614 \else
2615 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2616 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
2617 \def\@@gls@checkbar{\@gls@checkbar#3\null}%
2618 \fi
2619 \else
2620 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2621 \@gls@quotechar\@gls@encapchar}%
2622 \ifx\null#3\null
2623 \def\@@gls@checkbar{\@gls@checkbar#2||\null}%
2624 \else
2625 \def\@@gls@checkbar{\@gls@checkbar#2|#3\null}%
2626 \fi
2627 \fi
2628 \@@gls@checkbar
2629 }

```

\@gls@checklevel and for !:

```

2630 \def\@gls@checklevel#1!#2!#3\null{%
2631 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2632 \toks@={#1}%

```



```

2633 \ifx\null#2\null
2634 \ifx\null#3\null
2635 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2636 \def\@gls@checklevel{\relax}%
2637 \else
2638 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2639 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
2640 \def\@gls@checklevel{\@gls@checklevel#3\null}%
2641 \fi
2642 \else
2643 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2644 \@gls@quotechar\@gls@levelchar}%
2645 \ifx\null#3\null
2646 \def\@gls@checklevel{\@gls@checklevel#2!\null}%
2647 \else
2648 \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
2649 \fi
2650 \fi
2651 \@gls@checklevel
2652 }

```

\@gls@checkactual and for ?:

```

2653 \def\@gls@checkactual#1?#2?#3\null{%
2654 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2655 \toks@={#1}%
2656 \ifx\null#2\null
2657 \ifx\null#3\null
2658 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2659 \def\@gls@checkactual{\relax}%
2660 \else
2661 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2662 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
2663 \def\@gls@checkactual{\@gls@checkactual#3\null}%
2664 \fi
2665 \else
2666 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2667 \@gls@quotechar\@gls@actualchar}%
2668 \ifx\null#3\null
2669 \def\@gls@checkactual{\@gls@checkactual#2??\null}%
2670 \else
2671 \def\@gls@checkactual{\@gls@checkactual#2?#3\null}%
2672 \fi
2673 \fi
2674 \@gls@checkactual
2675 }

```

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

```

2676 \def\@gls@xdycheckquote#1"#2"#3\null{%
2677 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%

```

```

2678 \toks@={#1}%
2679 \ifx\null#2\null
2680 \ifx\null#3\null
2681 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2682 \def\@gls@xdycheckquote{\relax}%
2683 \else
2684 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2685 \string"\string"}%
2686 \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
2687 \fi
2688 \else
2689 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2690 \string"}%
2691 \ifx\null#3\null
2692 \def\@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
2693 \else
2694 \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
2695 \fi
2696 \fi
2697 \@gls@xdycheckquote
2698 }

```

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

```

2699 \edef\def\@gls@xdycheckbackslash{%
2700 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
2701 ##2\@backslashchar##3\noexpand\null{%
2702 \noexpand\@gls@tmpb=\noexpand\expandafter
2703 {\noexpand\@gls@checkedmkidx}%
2704 \noexpand\toks@={##1}%
2705 \noexpand\ifx\noexpand\null##2\noexpand\null
2706 \noexpand\ifx\noexpand\null##3\noexpand\null
2707 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2708 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
2709 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
2710 \noexpand\else
2711 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2712 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2713 \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
2714 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2715 \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
2716 \noexpand\fi
2717 \noexpand\else
2718 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2719 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2720 \@backslashchar\@backslashchar}%
2721 \noexpand\ifx\noexpand\null##3\noexpand\null
2722 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2723 \noexpand\@gls@xdycheckbackslash##2\@backslashchar

```

```

2724     \@backslashchar\noexpand\null}%
2725 \noexpand\else
2726     \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2727         \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2728             ##3\noexpand\null}%
2729 \noexpand\fi
2730 \noexpand\fi
2731 \noexpand\@gls@xdycheckbackslash
2732 }%
2733 }

```

Now go ahead and define \gls@xdycheckbackslash

```

2734 \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.

```

2735 \ifcsundef{hyperlink}%
2736 {%
2737     \gdef\@glslink#1#2{#2}%
2738 }%
2739 {%
2740     \gdef\@glslink#1#2{\hyperlink{#1}{#2}}%
2741 }

```

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

```

2742 \newlength\gls@tmplen \ifcsundef{hypertarget}%
2743 {%
2744     \gdef\@glstarget#1#2{#2}%
2745 }%
2746 {%
2747     \gdef\@glstarget#1#2{%
2748         \settoheight{\gls@tmplen}{#2}%
2749         \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
2750     }%
2751 }

```

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

\glsdisablehyper

```

2752 \newcommand{\glsdisablehyper}{%
2753     \renewcommand*\@glslink[2]{##2}%
2754     \renewcommand*\@glstarget[2]{##2}%
2755 }

```

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

`\glsenablehyper`

```
2756 \newcommand{\glsenablehyper}{%
2757 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
2758 \renewcommand*\@glsstar[2]{%
2759   \settoheight{\gls@tmplen}{##2}%
2760   \raisebox{\gls@tmplen}{\hypertarget{##1}{}}##2}}
```

Provide some convenience commands if not already defined:

```
2761 \providecommand{\@firstofthree}[3]{#1}
2762 \providecommand{\@secondofthree}[3]{#2}
2763 \providecommand{\@thirdofthree}[3]{#3}
```

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`

```
2764 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}
```

Define the starred form:

`\@sgls`

```
2765 \newcommand*\@sgls[1][\@gls[hyper=false,#1]]
```

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

`\@gls`

```
2766 \newcommand*\@gls[2][\@glsdoifexists{#2}]%
2767   \new@ifnextchar{\@gls@{#1}{#2}}{\@gls@{#1}{#2}[]}%
2768 }
```

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

```
2769 \def\@gls@#1#2[#3]{%
2770   \glsdoifexists{#2}%
2771   {%
2772     \edef\@gls@type{\glsentrytype{#2}}%
```

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

```

2773 \def\@gls@link@opts{#1}%
2774 \def\@gls@link@label{#2}%

2775 \def\glslabel{#2}%
2776 \let\glsifplural\@secondoftwo
2777 \let\glsupcase\@firstofthree
2778 \let\glsustomtext\@empty
2779 \def\glsinsert{#3}%

```

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

```

2780 \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%

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.

2781 \ifglsused{#2}%
2782 {%
2783 \@gls@link[#1]{#2}{\@glo@text}%
2784 }%
2785 {%
2786 \gls@checkisacronymlist\@glo@type
2787 \ifthenelse
2788 {(\boolean{glsisacronymlist}\AND \boolean{glsacrfootnote})\}
2789 \OR \NOT\boolean{glshyperfirst}
2790 }%
2791 {%
2792 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2793 }%
2794 {%
2795 \@gls@link[#1]{#2}{\@glo@text}%
2796 }%
2797 }%

```

Indicate that this entry has now been used

```

2798 \ifKV@glslink@local
2799 \glslocalunset{#2}%
2800 \else
2801 \glsunset{#2}%
2802 \fi
2803 }%
2804 }

```

\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

```

2805 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```
2806 \newcommand*{\@sGls}[1] [] {\@Gls[hyper=false,#1]}
```

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

```
2807 \newcommand*{\@Gls}[2] [] {%
2808   \new@ifnextchar[\@Gls@{#1}{#2}]{\@Gls@{#1}{#2} []}%
2809 }
```

\@Gls@ Read in the final optional argument:

```
2810 \def\@Gls@#1#2[#3]{%
2811   \glsdoifexists{#2}%
2812   {%
2813     \edef\@glo@type{\glsentrytype{#2}}%
2814     Save options in \@gls@link@opts and label in \@gls@link@label
2815     \def\@gls@link@opts{#1}%
2816     \def\@gls@link@label{#2}%
2817     \def\glslabel{#2}%
2818     \let\glsifplural\@secondoftwo
2819     \let\glsifscapscase\@secondofthree
2820     \let\glsifcustomtext\@empty
2821     \def\glsinsert{#3}%
```

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

```
2821     \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%
2822     Call \@gls@link If footnote package option has been used and the glossary
2823     type is \acronymtype, suppress hyperlink for first use. Likewise if the hyper-
2824     first=false package option is used.
2825     \ifglsused{#2}%
2826     {%
2827       \@gls@link[#1]{#2}{\@glo@text}%
2828     }%
2829     {%
2830       \gls@checkisacronymlist\@glo@type
2831       \ifthenelse
2832       {(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote})}
2833       {\OR \NOT\boolean{glshyperfirst}}%
2834       {\@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2835       }%
2836       {\@gls@link[#1]{#2}{\@glo@text}%
2837       }%
2838     }%
2839 }
```

Indicate that this entry has now been used

```

2840 \ifKV@glslink@local
2841 \glslocalunset{#2}%
2842 \else
2843 \glsunset{#2}%
2844 \fi
2845 }%
2846 }

```

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

\GLS

```

2847 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

2848 \newcommand*{\@sGLS}[1] [] {\@GLS[hyper=false,#1]}

```

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

```

2849 \newcommand*{\@GLS}[2] [] {%
2850 \new@ifnextchar[{\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}]%
2851 }

```

\@GLS@ Read in the final optional argument:

```

2852 \def\@GLS@#1#2[#3]{%
2853 \glsdoifexists{#2}%
2854 {%
2855 \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2856 \def\@gls@link@opts{#1}%
2857 \def\@gls@link@label{#2}%

2858 \def\glslabel{#2}%
2859 \let\glsifplural\@secondoftwo
2860 \let\glscapscase\@thirdofthree
2861 \let\glscustomtext\@empty
2862 \def\glsinsert{#3}%

```

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

```

2863 \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%

```

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.

```

2864 \ifglsused{#2}%
2865 {%
2866 \@gls@link[#1]{#2}{\@glo@text}%
2867 }%
2868 {%

```

```

2869 \gls@checkisacronymlist\@glo@type
2870 \ifthenelse
2871 {%
2872 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2873 \OR \NOT\boolean{glshyperfirst}}{%
2874 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2875 }%
2876 {%
2877 \@gls@link[#1]{#2}{\@glo@text}%
2878 }%
2879 }%

```

Indicate that this entry has now been used

```

2880 \ifKV@glslink@local
2881 \glslocalunset{#2}%
2882 \else
2883 \glsunset{#2}%
2884 \fi
2885 }%
2886 }

```

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

`\glspl`

```

2887 \newrobustcmd*{\glspl}{\@ifstar\@sglspl\@glspl}

```

Define the starred form:

```

2888 \newcommand*{\@sglspl}[1][\@glspl[hyper=false,#1]]{

```

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

```

2889 \newcommand*{\@glspl}[2][\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}
2890 \new@ifnextchar[\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}
2891 }

```

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

```

2892 \def\@glspl@#1#2[#3]{%
2893 \glsdoifexists{#2}%
2894 {%
2895 \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2896 \def\@gls@link@opts{#1}%
2897 \def\@gls@link@label{#2}%

2898 \def\glslabel{#2}%
2899 \let\glsifplural\@firstoftwo
2900 \let\glscapscase\@firstofthree
2901 \let\glscustomtext\@empty
2902 \def\glsinsert{#3}%
2903 % Determine what the link text should be (this is stored in

```



```

2904% \cs{@glo@text})
2905%\changes{1.12}{2008 Mar 8}{now uses \cs{glsentrydescplural} and
2906% \cs{glsentrysymbolplural} instead of \cs{glsentrydesc} and
2907% \cs{glsentrysymbol}}
2908%\changes{3.11a}{2013-10-15}{change to using \cs{glsentryfmt} style
2909%commands}
2910% \begin{macrocode}
2911 \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%

```

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.

```

2912 \ifglsused{#2}%
2913 {%
2914 \@gls@link[#1]{#2}{\@glo@text}%
2915 }%
2916 {%
2917 \gls@checkisacronymlist\@glo@type
2918 \ifthenelse
2919 {%
2920 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2921 \OR \NOT\boolean{glshyperfirst}%
2922 }%
2923 {%
2924 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2925 }%
2926 {%
2927 \@gls@link[#1]{#2}{\@glo@text}%
2928 }%
2929 }%

```

Indicate that this entry has now been used

```

2930 \ifKV@glslink@local
2931 \glslocalunset{#2}%
2932 \else
2933 \glsunset{#2}%
2934 \fi
2935 }%
2936 }

```

`\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`

```

2937 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```

2938 \newcommand*{\@sGlspl}[1][\@Glspl[hyper=false,#1]]{

```

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

```
2939 \newcommand*{\@Glspl}[2][]{%
2940   \new@ifnextchar[{\@Glspl@{#1}{#2}}{\@Glspl@{#1}{#2}[]}%
2941 }
```

\@Glspl@ Read in the final optional argument:

```
2942 \def\@Glspl@#1#2[#3]{%
2943   \glsdoifexists{#2}%
2944   {%
2945     \edef\@glo@type{\glsentrytype{#2}}%
2946     \def\@gls@link@opts{#1}%
2947     \def\@gls@link@label{#2}%
2948     \def\glslabel{#2}%
2949     \let\glsifplural\@firstoftwo
2950     \let\glsapscase\@secondofthree
2951     \let\glscustomtext\@empty
2952     \def\glsinsert{#3}%
2953     \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%
2954     \ifglsused{#2}%
2955       {%
2956         \@gls@link[#1]{#2}{\@glo@text}%
2957       }%
2958       {%
2959         \gls@checkisacronymlist\@glo@type
2960         \ifthenelse
2961           {\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\AND
2962            \OR \NOT\boolean{glshyperfirst}}%
2963           {\@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2964           }%
2965           {\@gls@link[#1]{#2}{\@glo@text}%
2966           }%
2967           {\@gls@link[#1]{#2}{\@glo@text}%
2968           }%
2969           {\@gls@link[#1]{#2}{\@glo@text}%
2970           }%
2971         }%
2972       }%
2973     }%
2974   }%
2975 }
```

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.

```
2953   \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%
2954   \ifglsused{#2}%
2955     {%
2956       \@gls@link[#1]{#2}{\@glo@text}%
2957     }%
2958     {%
2959       \gls@checkisacronymlist\@glo@type
2960       \ifthenelse
2961         {\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\AND
2962          \OR \NOT\boolean{glshyperfirst}}%
2963         {\@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2964         }%
2965         {\@gls@link[#1]{#2}{\@glo@text}%
2966         }%
2967         {\@gls@link[#1]{#2}{\@glo@text}%
2968         }%
2969         {\@gls@link[#1]{#2}{\@glo@text}%
2970         }%
2971       }%
2972     }%
2973   }%
2974 }
```

Indicate that this entry has now been used

```

2972 \ifKV@glslink@local
2973 \glsllocalunset{#2}%
2974 \else
2975 \glsunset{#2}%
2976 \fi
2977 }%
2978 }

```

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

`\GLSp1`

```

2979 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```

2980 \newcommand*{\@sGLSp1}[1][\@GLSp1[hyper=false,#1]]{

```

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

```

2981 \newcommand*{\@GLSp1}[2][\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}%
2982 \new@ifnextchar[\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}%
2983 }

```

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

```

2984 \def\@GLSp1@#1#2[#3]{%
2985 \glsoifexists{#2}%
2986 {%
2987 \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2988 \def\@gls@link@opts{#1}%
2989 \def\@gls@link@label{#2}%
2990 \def\@gls@label{#2}%
2991 \let\@gls@ifplural\@firstoftwo
2992 \let\@gls@scapscase\@thirdofthree
2993 \let\@gls@customtext\@empty
2994 \def\@gls@insert{#3}%

```

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

```

2995 \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%

```

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.

```

2996 \ifglsused{#2}%
2997 {%
2998 \@gls@link[#1]{#2}{\@glo@text}%
2999 }%

```

```

3000    {%
3001      \gls@checkisacronymlist\@glo@type
3002      \ifthenelse
3003      {%
3004        \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3005        \OR \NOT\boolean{glshyperfirst}}%
3006      }%
3007      {%
3008        \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3009      }%
3010      {%
3011        \@gls@link[#1]{#2}{\@glo@text}%
3012      }%
3013    }%

```

Indicate that this entry has now been used

```

3014      \ifKV@glslink@local
3015      \glslocalunset{#2}%
3016    \else
3017      \glsunset{#2}%
3018    \fi
3019  }%
3020 }

```

`\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:

```

3021 \newrobustcmd*{\glsdisp}{\@ifstar\sglsdisp\@glsdisp}

```

Define the starred form:

```

\@sgls
3022 \newcommand*{\@sglsdisp}[1][ ]{\@glsdisp[hyper=false,#1]}

```

Defined the un-starred form.

```

\@glsdisp
3023 \newcommand*{\@glsdisp}[3][ ]{%
3024   \glsdoifexists{#2}{%
3025     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

3026     \def\@gls@link@opts{#1}%
3027     \def\@gls@link@label{#2}%
3028     \def\glslabel{#2}%
3029     \let\glsifplural\@secondoftwo
3030     \let\glscapscase\@firstofthree
3031     \def\glscustomtext{#3}%
3032     \def\glsinsert{}%

```

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

```

3033 \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%
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.
3034 \ifglsused{#2}%
3035 {%
3036 \@gls@link[#1]{#2}{\@glo@text}%
3037 }%
3038 {%
3039 \gls@checkisacronymlist\@glo@type
3040 \ifthenelse{(\boolean{@glsisacronymlist}\AND
3041 \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
3042 {%
3043 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3044 }%
3045 {%
3046 \@gls@link[#1]{#2}{\@glo@text}%
3047 }%
3048 }%

```

Indicate that this entry has now been used

```

3049 \ifKV@glslink@local
3050 \glslocalunset{#2}%
3051 \else
3052 \glsunset{#2}%
3053 \fi
3054 }%
3055 }

```

\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

```

3056 \newrobustcmd*{\glstext}{\@ifstar\@sglstext\@glstext}

```

Define the starred form:

```

3057 \newcommand*{\@sglstext}[1][\@glstext[hyper=false,#1]]

```

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

```

3058 \newcommand*{\@glstext}[2][\@%
3059 \new@ifnextchar[\@glstext@{#1}{#2}]{\@glstext@{#1}{#2}[]}]

```

Read in the final optional argument:

```

3060 \def\@glstext@#1#2[#3]{%
3061 \glsdoifexists{#2}%
3062 {%
3063 \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3064   \@gls@link[#1]{#2}{\glsentrytext{#2}#3}%
3065   }%
3066 }

```

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

\GLStext

```

3067 \newrobustcmd*{\GLStext}{\@ifstar\@sGLStext\@GLStext}

```

Define the starred form:

```

3068 \newcommand*{\@sGLStext}[1] [] {\@GLStext[hyper=false,#1]}

```

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

```

3069 \newcommand*{\@GLStext}[2] [] {%
3070   \new@ifnextchar[{\@GLStext@{#1}{#2}}{\@GLStext@{#1}{#2} []}]

```

Read in the final optional argument:

```

3071 \def\@GLStext@#1#2[#3] {%
3072   \glsdoifexists{#2}%
3073   {%
3074     \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3075   \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%
3076   }%
3077 }

```

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

\Glsentrytext

```

3078 \newrobustcmd*{\Glsentrytext}{\@ifstar\@sGlsentrytext\@Glsentrytext}

```

Define the starred form:

```

3079 \newcommand*{\@sGlsentrytext}[1] [] {\@Glsentrytext[hyper=false,#1]}

```

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

```

3080 \newcommand*{\@Glsentrytext}[2] [] {%
3081   \new@ifnextchar[{\@Glsentrytext@{#1}{#2}}{\@Glsentrytext@{#1}{#2} []}]

```

Read in the final optional argument:

```

3082 \def\@Glsentrytext@#1#2[#3] {%
3083   \glsdoifexists{#2}%
3084   {%
3085     \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3086   \@gls@link[#1]{#2}{\Glsentrytext{#2}#3}%
3087   }%
3088 }

```

`\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`

```
3089 \newrobustcmd*{\glsfirst}{\@ifstar\sglsfirst\@glsfirst}
```

Define the starred form:

```
3090 \newcommand*{\sglsfirst}[1] [] {\@glsfirst[hyper=false,#1]}
```

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

```
3091 \newcommand*{\@glsfirst}[2] [] {%
```

```
3092   \new@ifnextchar[{\@glsfirst@{#1}{#2}}{\@glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3093 \def\@glsfirst@#1#2[#3] {%
```

```
3094   \glsdoifexists{#2}%
```

```
3095   {%
```

```
3096     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3097     \@gls@link[#1]{#2}{\glsentryfirst{#2}#3}%
```

```
3098   }%
```

```
3099 }
```

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

`\Glsfirst`

```
3100 \newrobustcmd*{\Glsfirst}{\@ifstar\sglsfirst\@Glsfirst}
```

Define the starred form:

```
3101 \newcommand*{\sglsfirst}[1] [] {\@Glsfirst[hyper=false,#1]}
```

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

```
3102 \newcommand*{\@Glsfirst}[2] [] {%
```

```
3103   \new@ifnextchar[{\@Glsfirst@{#1}{#2}}{\@Glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3104 \def\@Glsfirst@#1#2[#3] {%
```

```
3105   \glsdoifexists{#2}%
```

```
3106   {%
```

```
3107     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3108     \@gls@link[#1]{#2}{\Glsentryfirst{#2}#3}}%
```

```
3109 }
```

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

`\GLSfirst`

```
3110 \newrobustcmd*{\GLSfirst}{\@ifstar\sglsfirst\@GLSfirst}
```

Define the starred form:

```
3111 \newcommand*{\@sGLSfirst}[1] [] {\@GLSfirst[hyper=false,#1]}
```

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

```
3112 \newcommand*{\@GLSfirst}[2] [] {%
```

```
3113   \new@ifnextchar[\@GLSfirst@{#1}{#2}}{\@GLSfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3114 \def\@GLSfirst@#1#2[#3] {%
```

```
3115   \glsdoifexists{#2}%
```

```
3116   {%
```

```
3117     \edef\@glo@type{\glentrytype{#2}}%
```

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

```
3118   \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentryfirst{#2}#3}}%
```

```
3119   }%
```

```
3120 }
```

\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

```
3121 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
3122 \newcommand*{\@sglsplural}[1] [] {\@glsplural[hyper=false,#1]}
```

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

```
3123 \newcommand*{\@glsplural}[2] [] {%
```

```
3124   \new@ifnextchar[\@glsplural@{#1}{#2}}{\@glsplural@{#1}{#2} []}}
```

Read in the final optional argument:

```
3125 \def\@glsplural@#1#2[#3] {%
```

```
3126   \glsdoifexists{#2}%
```

```
3127   {%
```

```
3128     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3129   \@gls@link[#1]{#2}{\glentryplural{#2}#3}}%
```

```
3130   }%
```

```
3131 }
```

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

\Glsplural

```
3132 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}
```

Define the starred form:

```
3133 \newcommand*{\@sGlsplural}[1] [] {\@Glsplural[hyper=false,#1]}
```


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

```
3134 \newcommand*{\@Glsplural}[2] [] {%
3135   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3136 \def\@Glsplural@#1#2[#3] {%
3137   \glsdoifexists{#2}%
3138   {%
3139     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3140     \@gls@link[#1]{#2}{\glsentryplural{#2}{#3}}%
3141   }%
3142 }
```

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

\GLSplural

```
3143 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}
```

Define the starred form:

```
3144 \newcommand*{\@sGLSplural}[1] [] {\@GLSplural[hyper=false,#1]}
```

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

```
3145 \newcommand*{\@GLSplural}[2] [] {%
3146   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3147 \def\@GLSplural@#1#2[#3] {%
3148   \glsdoifexists{#2}%
3149   {%
3150     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3151     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}{#3}}}%
3152   }%
3153 }
```

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

\glsfirstplural

```
3154 \newrobustcmd*{\glsfirstplural}{\@ifstar\@sglsfirstplural\@glsfirstplural}
```

Define the starred form:

```
3155 \newcommand*{\@sglsfirstplural}[1] [] {\@glsfirstplural[hyper=false,#1]}
```

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

```
3156 \newcommand*{\@glsfirstplural}[2] [] {%
3157   \new@ifnextchar[{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```

3158 \def\@glsfirstplural@#1#2[#3]{%
3159   \glsdoifexists{#2}%
3160   {%
3161     \edef\@glo@type{\glsentrytype{#2}}%
3162     \@gls@link[#1]{#2}{\glsentryfirstplural{#2}#3}%
3163   }%
3164 }

```

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

\Glsfirstplural

```

3165 \newrobustcmd*{\Glsfirstplural}{\@ifstar\@sGlsfirstplural\@Glsfirstplural}

```

Define the starred form:

```

3166 \newcommand*{\@sGlsfirstplural}[1][\@Glsfirstplural[hyper=false,#1]]{

```

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

```

3167 \newcommand*{\@Glsfirstplural}[2][\@Glsfirstplural@{#1}{#2}[]]{%
3168   \new@ifnextchar[\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

3169 \def\@Glsfirstplural@#1#2[#3]{%
3170   \glsdoifexists{#2}%
3171   {%
3172     \edef\@glo@type{\glsentrytype{#2}}%
3173     \@gls@link[#1]{#2}{\Glsentryfirstplural{#2}#3}%
3174   }%
3175 }

```

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

\GLSfirstplural

```

3176 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\@GLSfirstplural}

```

Define the starred form:

```

3177 \newcommand*{\@sGLSfirstplural}[1][\@GLSfirstplural[hyper=false,#1]]{

```

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

```

3178 \newcommand*{\@GLSfirstplural}[2][\@GLSfirstplural@{#1}{#2}[]]{%
3179   \new@ifnextchar[\@GLSfirstplural@{#1}{#2}]{\@GLSfirstplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```
3180 \def\@GLSfirstplural@#1#2[#3]{%
3181   \glsdoifexists{#2}%
3182   {%
3183     \edef\@glo@type{\glsentrytype{#2}}%
3184     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryfirstplural{#2}#3}}%
3185   }%
3186 }
```

`\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`

```
3187 \newrobustcmd*{\glsname}{\@ifstar\@sglsname\@glsname}
```

Define the starred form:

```
3188 \newcommand*{\@sglsname}[1][]{\@glsname[hyper=false,#1]}
```

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

```
3189 \newcommand*{\@glsname}[2][]{%
3190   \new@ifnextchar[{\@glsname@{#1}{#2}}{\@glsname@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
3191 \def\@glsname@#1#2[#3]{%
3192   \glsdoifexists{#2}%
3193   {%
3194     \edef\@glo@type{\glsentrytype{#2}}%
3195     \@gls@link[#1]{#2}{\glsentryname{#2}#3}%
3196   }%
3197 }
```

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

`\Glsname`

```
3198 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}
```

Define the starred form:

```
3199 \newcommand*{\@sGlsname}[1][]{\@Glsname[hyper=false,#1]}
```

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

```
3200 \newcommand*{\@Glsname}[2][]{%
3201   \new@ifnextchar[{\@Glsname@{#1}{#2}}{\@Glsname@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
3202 \def\@Glsname@#1#2[#3]{%
3203   \glsdoifexists{#2}%
3204   {%
3205     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3206     \@gls@link[#1]{#2}{\Glsentryname{#2}#3}%
3207   }%
3208 }
```

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

\GLSname

```
3209 \newrobustcmd*{\@GLSname}{\@ifstar\@sGLSname\@GLSname}
```

Define the starred form:

```
3210 \newcommand*{\@sGLSname}[1][]{\@GLSname[hyper=false,#1]}
```

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

```
3211 \newcommand*{\@GLSname}[2][]{%
3212   \new@ifnextchar[{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
3213 \def\@GLSname@#1#2[#3]{%
3214   \glsdoifexists{#2}%
3215   {%
3216     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3217     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryname{#2}#3}}%
3218   }%
3219 }
```

\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

```
3220 \newrobustcmd*{\glsdesc}{\@ifstar\@sglsdesc\@glsdesc}
```

Define the starred form:

```
3221 \newcommand*{\@sglsdesc}[1][]{\@glsdesc[hyper=false,#1]}
```

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

```
3222 \newcommand*{\@glsdesc}[2][]{%
3223   \new@ifnextchar[{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
3224 \def\@glsdesc@#1#2[#3]{%
3225   \glsdoifexists{#2}%
3226   {%
3227     \edef\@glo@type{\glsentrytype{#2}}%
3228     \@gls@link[#1]{#2}{\glsentrydesc{#2}#3}%
3229   }%
3230 }
```

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

`\Glsdesc`

```
3231 \newrobustcmd*{\Glsdesc}{\@ifstar\@sGlsdesc\@Glsdesc}
```

Define the starred form:

```
3232 \newcommand*{\@sGlsdesc}[1][]{\@Glsdesc[hyper=false,#1]}
```

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

```
3233 \newcommand*{\@Glsdesc}[2][]{%
3234   \new@ifnextchar{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3235 \def\@Glsdesc@#1#2[#3]{%
3236   \glsdoifexists{#2}%
3237   {%
3238     \edef\@glo@type{\glsentrytype{#2}}%
3239     \@gls@link[#1]{#2}{\Glsentrydesc{#2}#3}%
3240   }%
3241 }
```

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

`\GLSdesc`

```
3242 \newrobustcmd*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}
```

Define the starred form:

```
3243 \newcommand*{\@sGLSdesc}[1][]{\@GLSdesc[hyper=false,#1]}
```

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

```
3244 \newcommand*{\@GLSdesc}[2][]{%
3245   \new@ifnextchar{\@GLSdesc@{#1}{#2}}{\@GLSdesc@{#1}{#2}[]}}
```

Read in the final optional argument:

```

3246 \def\@GLSdesc@#1#2[#3]{%
3247   \glsdoifexists{#2}%
3248   {%
3249     \edef\@glo@type{\glsentrytype{#2}}%
3250     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentrydesc{#2}#3}}%
3251   }%
3252 }

```

Call \@gls@link

\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

```

3253 \newrobustcmd*{\glsdescplural}{\@ifstar\@sglsdescplural\@glsdescplural}

```

Define the starred form:

```

3254 \newcommand*{\@sglsdescplural}[1][\@glsdescplural[hyper=false,#1]]{

```

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

```

3255 \newcommand*{\@glsdescplural}[2][\@glsdescplural@{#1}{#2}[]]{%
3256   \new@ifnextchar[\@glsdescplural@{#1}{#2}]{\@glsdescplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

3257 \def\@glsdescplural@#1#2[#3]{%
3258   \glsdoifexists{#2}%
3259   {%
3260     \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3261   \@gls@link[#1]{#2}{\glsentrydescplural{#2}#3}%
3262   }%
3263 }

```

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

\Glsdescplural

```

3264 \newrobustcmd*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}

```

Define the starred form:

```

3265 \newcommand*{\@sGlsdescplural}[1][\@Glsdescplural[hyper=false,#1]]{

```

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

```

3266 \newcommand*{\@Glsdescplural}[2][\@Glsdescplural@{#1}{#2}[]]{%
3267   \new@ifnextchar[\@Glsdescplural@{#1}{#2}]{\@Glsdescplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```
3268 \def\@GLSdescplural@#1#2[#3]{%
3269   \glsdoifexists{#2}%
3270   {%
3271     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3272     \@gls@link[#1]{#2}{\glsentrydescplural{#2}#3}%
3273   }%
3274 }
```

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

\GLSdescplural

```
3275 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}
```

Define the starred form:

```
3276 \newcommand*{\@sGLSdescplural}[1][\@GLSdescplural[hyper=false,#1]]
```

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

```
3277 \newcommand*{\@GLSdescplural}[2][\@GLSdescplural@{#1}{#2}[]]{%
3278   \new@ifnextchar{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2}[]]{}
```

Read in the final optional argument:

```
3279 \def\@GLSdescplural@#1#2[#3]{%
3280   \glsdoifexists{#2}%
3281   {%
3282     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3283     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentrydescplural{#2}#3}}%
3284   }%
3285 }
```

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

\glssymbol

```
3286 \newrobustcmd*{\glssymbol}{\@ifstar\@sglssymbol\@glssymbol}
```

Define the starred form:

```
3287 \newcommand*{\@sglssymbol}[1][\@glssymbol[hyper=false,#1]]
```

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

```
3288 \newcommand*{\@glssymbol}[2][\@glssymbol@{#1}{#2}[]]{%
3289   \new@ifnextchar{\@glssymbol@{#1}{#2}}{\@glssymbol@{#1}{#2}[]]{}
```

Read in the final optional argument:

```
3290 \def\@glssymbol@#1#2[#3]{%
3291   \glsdoifexists{#2}%
3292   {%
3293     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3294     \@gls@link[#1]{#2}{\glentrysymbol{#2}#3}%
3295   }%
3296 }
```

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

\Glsymbol

```
3297 \newrobustcmd*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}
```

Define the starred form:

```
3298 \newcommand*{\@sGlsymbol}[1][]{\@Glsymbol[hyper=false,#1]}
```

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

```
3299 \newcommand*{\@Glsymbol}[2][]{%
3300   \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
3301 \def\@Glsymbol@#1#2[#3]{%
3302   \glsdoifexists{#2}%
3303   {%
3304     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3305     \@gls@link[#1]{#2}{\glentrysymbol{#2}#3}%
3306   }%
3307 }
```

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

\GLSsymbol

```
3308 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}
```

Define the starred form:

```
3309 \newcommand*{\@sGLSsymbol}[1][]{\@GLSsymbol[hyper=false,#1]}
```

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

```
3310 \newcommand*{\@GLSsymbol}[2][]{%
3311   \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2}[]}]}
```


Read in the final optional argument:

```

3312 \def\@GLSsymbol@#1#2[#3]{%
3313   \glsdoifexists{#2}%
3314   {%
3315     \edef\@glo@type{\glstrytype{#2}}%
3316     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glstrysymbol{#2}#3}}%
3317   }%
3318 }

```

`\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`

```

3319 \newrobustcmd*{\glssymbolplural}{\@ifstar\@sglssymbolplural\glssymbolplural}

```

Define the starred form:

```

3320 \newcommand*{\@sglssymbolplural}[1][\@glssymbolplural[hyper=false,#1]]{

```

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

```

3321 \newcommand*{\@glssymbolplural}[2][\@glssymbolplural@#1]{%
3322   \new@ifnextchar{\@glssymbolplural@{#1}{#2}}{\@glssymbolplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

3323 \def\@glssymbolplural@#1#2[#3]{%
3324   \glsdoifexists{#2}%
3325   {%
3326     \edef\@glo@type{\glstrytype{#2}}%
3327     \@gls@link[#1]{#2}{\glstrysymbolplural{#2}#3}%
3328   }%
3329 }

```

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

`\Glsymbolplural`

```

3330 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\Glsymbolplural}

```

Define the starred form:

```

3331 \newcommand*{\@sGlsymbolplural}[1][\@Glsymbolplural[hyper=false,#1]]{

```

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

```

3332 \newcommand*{\@Glsymbolplural}[2][\@Glsymbolplural@#1]{%
3333   \new@ifnextchar{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

3334 \def\@Glsymbolplural@#1#2[#3]{%
3335   \glsdoifexists{#2}%
3336   {%
3337     \edef\@glo@type{\glentrytype{#2}}%
3338     \@gls@link[#1]{#2}{\Glsentrysymbolplural{#2}#3}%
3339   }%
3340 }

```

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

`\GLSsymbolplural`

```

3341 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\GLSsymbolplural}

```

Define the starred form:

```

3342 \newcommand*{\@sGLSsymbolplural}[1][\@GLSsymbolplural[hyper=false,#1]]

```

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

```

3343 \newcommand*{\@GLSsymbolplural}[2][\@GLSsymbolplural@#1]{%
3344   \new@ifnextchar[\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

3345 \def\@GLSsymbolplural@#1#2[#3]{%
3346   \glsdoifexists{#2}%
3347   {%
3348     \edef\@glo@type{\glentrytype{#2}}%
3349     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentrysymbolplural{#2}#3}}%
3350   }%
3351 }

```

`\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`

```

3352 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\glsuseri}

```

Define the starred form:

```

3353 \newcommand*{\@sglsuseri}[1][\@glsuseri[hyper=false,#1]]

```

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

```

3354 \newcommand*{\@glsuseri}[2][\@glsuseri@#1]{%
3355   \new@ifnextchar[\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2}[]}}

```

Read in the final optional argument:

```

3356 \def\@glsuseri@#1#2[#3]{%
3357   \glsdoifexists{#2}%
3358   {%
3359     \edef\@glo@type{\glentrytype{#2}}%

```

Call \@gls@link

```

3360 \@gls@link[#1]{#2}{\glsentryuseri{#2}#3}%
3361 }%
3362 }

```

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

\Glsuseri

```

3363 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}

```

Define the starred form:

```

3364 \newcommand*{\@sGlsuseri}[1][\@Glsuseri[hyper=false,#1]]{

```

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

```

3365 \newcommand*{\@Glsuseri}[2][\@Glsuseri@{#1}{#2}]{\@Glsuseri@{#1}{#2}[]}
3366 \new@ifnextchar[\@Glsuseri@{#1}{#2}]{\@Glsuseri@{#1}{#2}[]}

```

Read in the final optional argument:

```

3367 \def\@Glsuseri@#1#2[#3]{%
3368 \glsdoifexists{#2}%
3369 {%
3370 \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3371 \@gls@link[#1]{#2}{\glsentryuseri{#2}#3}%
3372 }%
3373 }

```

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

\GLSuseri

```

3374 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}

```

Define the starred form:

```

3375 \newcommand*{\@sGLSuseri}[1][\@GLSuseri[hyper=false,#1]]{

```

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

```

3376 \newcommand*{\@GLSuseri}[2][\@GLSuseri@{#1}{#2}]{\@GLSuseri@{#1}{#2}[]}
3377 \new@ifnextchar[\@GLSuseri@{#1}{#2}]{\@GLSuseri@{#1}{#2}[]}

```

Read in the final optional argument:

```

3378 \def\@GLSuseri@#1#2[#3]{%
3379 \glsdoifexists{#2}%
3380 {%
3381 \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3382 \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
3383 }%
3384 }

```

`\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`

```
3385 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}
```

Define the starred form:

```
3386 \newcommand*{\@sglsuserii}[1] [] {\@glsuserii[hyper=false,#1]}
```

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

```
3387 \newcommand*{\@glsuserii}[2] [] {%
```

```
3388   \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3389 \def\@glsuserii@#1#2[#3] {%
```

```
3390   \glsdoifexists{#2}%
```

```
3391   {%
```

```
3392     \edef\@glo@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3393     \@gls@link[#1]{#2}{\glsentryuserii{#2}#3}%
```

```
3394   }%
```

```
3395 }
```

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

`\Glsuserii`

```
3396 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
3397 \newcommand*{\@sGlsuserii}[1] [] {\@Glsuserii[hyper=false,#1]}
```

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

```
3398 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3399   \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3400 \def\@Glsuserii@#1#2[#3] {%
```

```
3401   \glsdoifexists{#2}%
```

```
3402   {%
```

```
3403     \edef\@glo@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3404     \@gls@link[#1]{#2}{\Glsentryuserii{#2}#3}%
```

```
3405   }%
```

```
3406 }
```

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

\GLSuserii

```
3407 \newrobustcmd*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}
```

Define the starred form:

```
3408 \newcommand*{\@sGLSuserii}[1] [] {\@GLSuserii[hyper=false,#1]}
```

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

```
3409 \newcommand*{\@GLSuserii}[2] [] {%
```

```
3410   \new@ifnextchar[{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3411 \def\@GLSuserii@#1#2[#3] {%
```

```
3412   \glsdoifexists{#2}%
```

```
3413   {%
```

```
3414     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3415     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
```

```
3416   }%
```

```
3417 }
```

\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

```
3418 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}
```

Define the starred form:

```
3419 \newcommand*{\@sglsuseriii}[1] [] {\@glsuseriii[hyper=false,#1]}
```

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

```
3420 \newcommand*{\@glsuseriii}[2] [] {%
```

```
3421   \new@ifnextchar[{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3422 \def\@glsuseriii@#1#2[#3] {%
```

```
3423   \glsdoifexists{#2}%
```

```
3424   {%
```

```
3425     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3426     \@gls@link[#1]{#2}{\glsentryuseriii{#2}#3}%
```

```
3427   }%
```

```
3428 }
```

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

\Glsuseriii

```
3429 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}
```

Define the starred form:

```
3430 \newcommand*{\@sGlsuseriii}[1] [] {\@Glsuseriii[hyper=false,#1]}
```

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

```
3431 \newcommand*{\@Glsuseriii}[2] [] {%
```

```
3432   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}]{\@Glsuseriii@{#1}{#2} []}}
```

Read in the final optional argument:

```
3433 \def\@Glsuseriii@#1#2[#3]{%
```

```
3434   \glsdoifexists{#2}%
```

```
3435   {%
```

```
3436     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3437     \@gls@link[#1]{#2}{\glsentryuseriii{#2}#3}%
```

```
3438   }%
```

```
3439 }
```

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

\GLSuseriii

```
3440 \newrobustcmd*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}
```

Define the starred form:

```
3441 \newcommand*{\@sGLSuseriii}[1] [] {\@GLSuseriii[hyper=false,#1]}
```

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

```
3442 \newcommand*{\@GLSuseriii}[2] [] {%
```

```
3443   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}]{\@GLSuseriii@{#1}{#2} []}}
```

Read in the final optional argument:

```
3444 \def\@GLSuseriii@#1#2[#3]{%
```

```
3445   \glsdoifexists{#2}%
```

```
3446   {%
```

```
3447     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3448     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseriii{#2}#3}}%
```

```
3449   }%
```

```
3450 }
```

\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

```
3451 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}
```

Define the starred form:

```
3452 \newcommand*{\@sglsuseriv}[1] [] {\@glsuseriv[hyper=false,#1]}
```

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

```
3453 \newcommand*{\@glsuseriv}[2] [] {%
3454   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3455 \def\@glsuseriv@#1#2[#3] {%
3456   \glsdoifexists{#2}%
3457   {%
3458     \edef\@glo@type{\glsentrytype{#2}}%
3459 % Call \cs{@gls@link}
3460 %\changes{3.11a}{2013-10-15}{changed to just use \cs{glsentryuseriv}}
3461 %   \begin{macrocode}
3462   \@gls@link[#1]{#2}{\glsentryuseriv{#2}#3}%
3463   }%
3464 }
```

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

`\Glsuseriv`

```
3465 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
3466 \newcommand*{\@sGlsuseriv}[1] [] {\@Glsuseriv[hyper=false,#1]}
```

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

```
3467 \newcommand*{\@Glsuseriv}[2] [] {%
3468   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3469 \def\@Glsuseriv@#1#2[#3] {%
3470   \glsdoifexists{#2}%
3471   {%
3472     \edef\@glo@type{\glsentrytype{#2}}%
3473     Call \@gls@link
3474     \@gls@link[#1]{#2}{\Glsentryuseriv{#2}#3}%
3475     }%
```

Call `\@gls@link`

```
3473   \@gls@link[#1]{#2}{\Glsentryuseriv{#2}#3}%
3474   }%
3475 }
```

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

`\GLSuseriv`

```
3476 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}
```

Define the starred form:

```
3477 \newcommand*{\@sGLSuseriv}[1] [] {\@GLSuseriv[hyper=false,#1]}
```

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

```
3478 \newcommand*{\@GLSuseriv}[2] [] {%
3479   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3480 \def\@GLSuseriv@#1#2[#3] {%
3481   \glsdoifexists{#2}%
3482   {%
3483     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3484   \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseriv{#2}#3}}%
3485   }%
3486 }
```

\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

```
3487 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}
```

Define the starred form:

```
3488 \newcommand*{\@sglsuserv}[1] [] {\@glsuserv[hyper=false,#1]}
```

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

```
3489 \newcommand*{\@glsuserv}[2] [] {%
3490   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3491 \def\@glsuserv@#1#2[#3] {%
3492   \glsdoifexists{#2}%
3493   {%
3494     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3495   \@gls@link[#1]{#2}{\glsentryuseriv{#2}#3}%
3496   }%
3497 }
```

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

\Glsuserv

```
3498 \newrobustcmd*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}
```

Define the starred form:

```
3499 \newcommand*{\@sGlsuserv}[1] [] {\@Glsuserv[hyper=false,#1]}
```

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

```
3500 \newcommand*{\@Glsuserv}[2] [] {%
3501   \new@ifnextchar[{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2} [] }}
```


Read in the final optional argument:

```
3502 \def\@Glsuserv@#1#2[#3]{%
3503   \glsdoifexists{#2}%
3504   {%
3505     \edef\@gls@type{\glsentrytype{#2}}%
3506     \@gls@link[#1]{#2}{\glsentryuserv{#2}#3}%
3507   }%
3508 }
```

Call `\@gls@link`

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

`\Glsuserv`

```
3509 \newrobustcmd*{\@Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}
```

Define the starred form:

```
3510 \newcommand*{\@sGlsuserv}[1][\@Glsuserv[hyper=false,#1]]{
```

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

```
3511 \newcommand*{\@Glsuserv}[2][\@Glsuserv@{#1}{#2}[]]{%
3512 \new@ifnextchar[\@Glsuserv@{#1}{#2}]{\@Glsuserv@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3513 \def\@Glsuserv@#1#2[#3]{%
3514   \glsdoifexists{#2}%
3515   {%
3516     \edef\@gls@type{\glsentrytype{#2}}%
3517     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuserv{#2}#3}}%
3518   }%
3519 }
```

`\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`

```
3520 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}
```

Define the starred form:

```
3521 \newcommand*{\@sglsuservi}[1][\@glsuservi[hyper=false,#1]]{
```

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

```
3522 \newcommand*{\@glsuservi}[2][\@glsuservi@{#1}{#2}[]]{%
3523 \new@ifnextchar[\@glsuservi@{#1}{#2}]{\@glsuservi@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3524 \def\@glsuservi@#1#2[#3]{%
3525   \glsdoifexists{#2}%
3526   {%
3527     \edef\@glo@type{\glsentrytype{#2}}%
3528     \@gls@link[#1]{#2}{\glsentryuservi{#2}#3}%
3529   }%
3530 }
```

Call \@gls@link

\Glsuservi behaves like \glsuservi except that the first letter is converted to uppercase.

\Glsuservi

```
3531 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}
```

Define the starred form:

```
3532 \newcommand*{\@sGlsuservi}[1][]{\@Glsuservi[hyper=false,#1]}
```

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

```
3533 \newcommand*{\@Glsuservi}[2][]{%
3534   \new@ifnextchar[\@Glsuservi@{#1}{#2}]{\@Glsuservi@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3535 \def\@Glsuservi@#1#2[#3]{%
3536   \glsdoifexists{#2}%
3537   {%
3538     \edef\@glo@type{\glsentrytype{#2}}%
3539     \@gls@link[#1]{#2}{\Glsentryuservi{#2}#3}%
3540   }%
3541 }
```

Call \@gls@link

\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

```
3542 \newrobustcmd*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}
```

Define the starred form:

```
3543 \newcommand*{\@sGLSuservi}[1][]{\@GLSuservi[hyper=false,#1]}
```

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

```
3544 \newcommand*{\@GLSuservi}[2][]{%
3545   \new@ifnextchar[\@GLSuservi@{#1}{#2}]{\@GLSuservi@{#1}{#2}[]}}
```

Read in the final optional argument:

```

3546 \def\@GLSuservi@#1#2[#3]{%
3547   \glsdoifexists{#2}%
3548   {%
3549     \edef\@glo@type{\glsentrytype{#2}}%

    Call \@gls@link
3550     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuservi{#2}#3}}%
3551   }%
3552 }
```

Now deal with acronym related keys. First the short form:

`\acrshort`

```

3553 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\@ns@acrshort}
```

Define the starred form:

```

3554 \newcommand*{\s@acrshort}[2][ ]{%
3555   \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}%
3556   {\@acrshort{hyper=false,#1}{#2}[ ]}%
3557 }
```

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

```

3558 \newcommand*{\ns@acrshort}[2][ ]{%
3559   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2}[ ]}%
3560 }
```

Read in the final optional argument:

```

3561 \def\@acrshort#1#2[#3]{%
3562   \glsdoifexists{#2}%
3563   {%
3564     \edef\@glo@type{\glsentrytype{#2}}%

3565     \def\glslabel{#2}%
3566     \let\glsifplural\@secondoftwo
3567     \let\glscapscase\@firstofthree
3568     \let\glsinsert\@empty
3569     \def\glscustomtext{%
3570       \acronymfont{\glsentryshort{#2}}#3%
3571     }%
```

Call \@gls@link

```

3572     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3573   }%
3574 }
```

`\Acrshort`

```

3575 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\@ns@Acrshort}
```

Define the starred form:

```
3576 \newcommand*{\s@Acrshort}[2] [] {%
3577   \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}]{%
3578     {\@Acrshort{hyper=false,#1}{#2} []}%
3579 }
```

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

```
3580 \newcommand*{\ns@Acrshort}[2] [] {%
3581   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}%
3582 }
```

Read in the final optional argument:

```
3583 \def\@Acrshort#1#2[#3] {%
3584   \glsdoifexists{#2}%
3585   {%
3586     \edef\@glo@type{\glsentrytype{#2}}%

3587     \def\glslabel{#2}%
3588     \let\glsifplural\@secondoftwo
3589     \let\glsupcase\@secondofthree
3590     \let\glsinsert\@empty
3591     \def\glscustomtext{%
3592       \acronymfont{\Glsentryshort{#2}}#3%
3593     }%
```

Call \@gls@link

```
3594   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3595   }%
3596 }
```

\ACRshort

```
3597 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}
```

Define the starred form:

```
3598 \newcommand*{\s@ACRshort}[2] [] {%
3599   \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}]{%
3600     {\@ACRshort{hyper=false,#1}{#2} []}%
3601 }
```

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

```
3602 \newcommand*{\ns@ACRshort}[2] [] {%
3603   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
3604 }
```

Read in the final optional argument:

```
3605 \def\@ACRshort#1#2[#3] {%
3606   \glsdoifexists{#2}%
3607   {%
3608     \edef\@glo@type{\glsentrytype{#2}}%
```

```

3609 \def\glslabel{#2}%
3610 \let\glsifplural\@secondoftwo
3611 \let\glscapscase\@thirdofthree
3612 \let\glsinsert\@empty
3613 \def\glscustomtext{%
3614     \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3615 }%

Call \@gls@link
3616 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3617 }%
3618 }

```

Short plural:

\acrshortpl

```

3619 \newrobustcmd*{\acrshortpl}{\ifstar\s@acrshortpl\ns@acrshortpl}

```

Define the starred form:

```

3620 \newcommand*{\s@acrshortpl}[2] [] {%
3621     \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}%
3622         {\@acrshortpl{hyper=false,#1}{#2} []}%
3623 }

```

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

```

3624 \newcommand*{\ns@acrshortpl}[2] [] {%
3625     \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
3626 }

```

Read in the final optional argument:

```

3627 \def\@acrshortpl#1#2[#3] {%
3628     \glsdoifexists{#2}%
3629     {%
3630         \edef\@glo@type{\glsentrytype{#2}}%

3631         \def\glslabel{#2}%
3632         \let\glsifplural\@firstoftwo
3633         \let\glscapscase\@firstofthree
3634         \let\glsinsert\@empty
3635         \def\glscustomtext{%
3636             \acronymfont{\glsentryshortpl{#2}}#3%
3637         }%

```

Call \@gls@link

```

3638     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3639 }%
3640 }

```

\Acrshortpl

```

3641 \newrobustcmd*{\Acrshortpl}{\ifstar\s@Acrshortpl\ns@Acrshortpl}

```

Define the starred form:

```
3642 \newcommand*{\s@Acrshortpl}[2] [] {%
3643   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}}%
3644   {\@Acrshortpl{hyper=false,#1}{#2} []}%
3645 }
```

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

```
3646 \newcommand*{\ns@Acrshortpl}[2] [] {%
3647   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
3648 }
```

Read in the final optional argument:

```
3649 \def\@Acrshortpl#1#2[#3] {%
3650   \glsdoifexists{#2}%
3651   {%
3652     \edef\@glo@type{\glsentrytype{#2}}%

3653     \def\glslabel{#2}%
3654     \let\glsifplural\@firstoftwo
3655     \let\glsupcase\@secondofthree
3656     \let\glsinsert\@empty
3657     \def\glscustomtext{%
3658       \acronymfont{\Glsentryshortpl{#2}}#3%
3659     }%
```

Call \@gls@link

```
3660   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3661   }%
3662 }
```

\ACRshortpl

```
3663 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}
```

Define the starred form:

```
3664 \newcommand*{\s@ACRshortpl}[2] [] {%
3665   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}}%
3666   {\@ACRshortpl{hyper=false,#1}{#2} []}%
3667 }
```

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

```
3668 \newcommand*{\ns@ACRshortpl}[2] [] {%
3669   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2} []}%
3670 }
```

Read in the final optional argument:

```
3671 \def\@ACRshortpl#1#2[#3] {%
3672   \glsdoifexists{#2}%
3673   {%
3674     \edef\@glo@type{\glsentrytype{#2}}%
```

```

3675 \def\glslabel{#2}%
3676 \let\glsifplural\@firstoftwo
3677 \let\glscapscase\@thirdofthree
3678 \let\glsinsert\@empty
3679 \def\glscustomtext{%
3680 \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}%
3681 }%

Call \@gls@link
3682 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3683 }%
3684 }

```

\acrlong

```

3685 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\@ns@acrlong}

```

Define the starred form:

```

3686 \newcommand*{\s@acrlong}[2][]{%
3687 \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}%
3688 {\@acrlong{hyper=false,#1}{#2}[]}%
3689 }

```

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

```

3690 \newcommand*{\ns@acrlong}[2][]{%
3691 \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2}[]}%
3692 }

```

Read in the final optional argument:

```

3693 \def\@acrlong#1#2[#3]{%
3694 \glsdoifexists{#2}%
3695 {%
3696 \edef\@glo@type{\glsentrytype{#2}}%

3697 \def\glslabel{#2}%
3698 \let\glsifplural\@secondoftwo
3699 \let\glscapscase\@firstofthree
3700 \let\glsinsert\@empty
3701 \def\glscustomtext{%
3702 \acronymfont{\glsentrylong{#2}}#3%
3703 }%

```

Call \@gls@link

```

3704 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3705 }%
3706 }

```

\Acrlong

```

3707 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\@ns@Acrlong}

```

Define the starred form:

```
3708 \newcommand*{\s@Acrlong}[2] [] {%
3709   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}}%
3710   {\@Acrlong{hyper=false,#1}{#2} []}%
3711 }
```

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

```
3712 \newcommand*{\ns@Acrlong}[2] [] {%
3713   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
3714 }
```

Read in the final optional argument:

```
3715 \def\@Acrlong#1#2[#3] {%
3716   \glsdoifexists{#2}%
3717   {%
3718     \edef\@glo@type{\glsentrytype{#2}}%

3719     \def\glslabel{#2}%
3720     \let\glsifplural\@secondoftwo
3721     \let\gls caps case\@secondofthree
3722     \let\glsinsert\@empty
3723     \def\gls custom text{%
3724       \acronymfont{\Glsentrylong{#2}}#3%
3725     }%
```

Call \@gls@link

```
3726   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3727   }%
3728 }
```

\ACRlong

```
3729 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\ns@ACRlong}
```

Define the starred form:

```
3730 \newcommand*{\s@ACRlong}[2] [] {%
3731   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}}%
3732   {\@ACRlong{hyper=false,#1}{#2} []}%
3733 }
```

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

```
3734 \newcommand*{\ns@ACRlong}[2] [] {%
3735   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
3736 }
```

Read in the final optional argument:

```
3737 \def\@ACRlong#1#2[#3] {%
3738   \glsdoifexists{#2}%
3739   {%
3740     \edef\@glo@type{\glsentrytype{#2}}%
```



```

3741 \def\glslabel{#2}%
3742 \let\glsifplural\@secondoftwo
3743 \let\glscapscase\@thirdofthree
3744 \let\glsinsert\@empty
3745 \def\glscustomtext{%
3746     \mfirstucMakeUppercase{\acronymfont{\glsentrylong{#2}}#3}%
3747 }%

Call \@gls@link
3748 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3749 }%
3750 }

```

Short plural:

\acrlongpl

```

3751 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}

```

Define the starred form:

```

3752 \newcommand*{\s@acrlongpl}[2] [] {%
3753     \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
3754         {\@acrlongpl{hyper=false,#1}{#2} []}%
3755 }

```

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

```

3756 \newcommand*{\ns@acrlongpl}[2] [] {%
3757     \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
3758 }

```

Read in the final optional argument:

```

3759 \def\@acrlongpl#1#2[#3] {%
3760     \glsdoifexists{#2}%
3761     {%
3762         \edef\@glo@type{\glsentrytype{#2}}%

3763         \def\glslabel{#2}%
3764         \let\glsifplural\@firstoftwo
3765         \let\glscapscase\@firstofthree
3766         \let\glsinsert\@empty
3767         \def\glscustomtext{%
3768             \acronymfont{\glsentrylongpl{#2}}#3%
3769         }%

```

Call \@gls@link

```

3770 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3771 }%
3772 }

```

\Acrlongpl

```

3773 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}

```

Define the starred form:

```
3774 \newcommand*{\s@Acrlongpl}[2] [] {%
3775   \new@ifnextchar[{\@Acrlongpl{hyper=false,#1}{#2}}}%
3776   {\@Acrlongpl{hyper=false,#1}{#2} []}%
3777 }
```

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

```
3778 \newcommand*{\ns@Acrlongpl}[2] [] {%
3779   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%
3780 }
```

Read in the final optional argument:

```
3781 \def\@Acrlongpl#1#2[#3] {%
3782   \glsdoifexists{#2}%
3783   {%
3784     \edef\@glo@type{\glsentrytype{#2}}%

3785     \def\glslabel{#2}%
3786     \let\glsifplural\@firstoftwo
3787     \let\glscapscase\@secondofthree
3788     \let\glsinsert\@empty
3789     \def\glscustomtext{%
3790       \acronymfont{\Glsentrylongpl{#2}}#3%
3791     }%
```

Call \@gls@link

```
3792   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3793   }%
3794 }
```

\ACRlongpl

```
3795 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}
```

Define the starred form:

```
3796 \newcommand*{\s@ACRlongpl}[2] [] {%
3797   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}}%
3798   {\@ACRlongpl{hyper=false,#1}{#2} []}%
3799 }
```

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

```
3800 \newcommand*{\ns@ACRlongpl}[2] [] {%
3801   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3802 }
```

Read in the final optional argument:

```
3803 \def\@ACRlongpl#1#2[#3] {%
3804   \glsdoifexists{#2}%
3805   {%
3806     \edef\@glo@type{\glsentrytype{#2}}%
```

```

3807 \def\glslabel{#2}%
3808 \let\glsifplural\@firstoftwo
3809 \let\glscapscase\@thirdofthree
3810 \let\glsinsert\@empty
3811 \def\glscustomtext{%
3812     \mfirstucMakeUppercase{\acronymfont{\glsentrylongpl{#2}}#3}%
3813 }%

    Call \@gls@link
3814 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3815 }%
3816 }

```

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
3817 \newcommand*{\glsentryname}[1]{\csname glo@#1@name\endcsname}

\Glsentryname
3818 \newrobustcmd*{\Glsentryname}[1]{%
3819 \protected@edef\@glo@text{\csname glo@#1@name\endcsname}%
3820 \expandafter\makefirstuc\expandafter{\@glo@text}%
3821 }

```

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
3822 \newcommand*{\glsentrydesc}[1]{\csname glo@#1@desc\endcsname}

\Glsentrydesc
3823 \newrobustcmd*{\Glsentrydesc}[1]{%
3824 \protected@edef\@glo@text{\csname glo@#1@desc\endcsname}%
3825 \expandafter\makefirstuc\expandafter{\@glo@text}%
3826 }

```

Plural form:

`\glentrydescplural`

```
3827 \newcommand*{\glentrydescplural}[1]{%
3828   \csname glo@#1@descplural\endcsname}
```

`\Glsentrydescplural`

```
3829 \newrobustcmd*{\Glsentrydescplural}[1]{%
3830 \protected@edef\@glo@text{\csname glo@#1@descplural\endcsname}%
3831 \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`

```
3832 \newcommand*{\glentrytext}[1]{\csname glo@#1@text\endcsname}
```

`\Glsentrytext`

```
3833 \newrobustcmd*{\Glsentrytext}[1]{%
3834   \protected@edef\@glo@text{\csname glo@#1@text\endcsname}%
3835   \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form:

`\glentryplural`

```
3836 \newcommand*{\glentryplural}[1]{\csname glo@#1@plural\endcsname}
```

`\Glsentryplural`

```
3837 \newrobustcmd*{\Glsentryplural}[1]{%
3838 \protected@edef\@glo@text{\csname glo@#1@plural\endcsname}%
3839 \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`

```
3840 \newcommand*{\glentrysymbol}[1]{\csname glo@#1@symbol\endcsname}
```

`\Glsentrysymbol`

```
3841 \newrobustcmd*{\Glsentrysymbol}[1]{%
3842 \protected@edef\@glo@text{\csname glo@#1@symbol\endcsname}%
3843 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glentrysymbolplural`

```
3844 \newcommand*{\glentrysymbolplural}[1]{%
3845   \csname glo@#1@symbolplural\endcsname}
```

lentrysymbolplural

```
3846 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3847 \protected@edef\@glo@text{\csname glo@#1@symbolplural\endcsname}%
3848 \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

```
3849 \newcommand*{\glsentryfirst}[1]{\csname glo@#1@first\endcsname}
```

\Glsentryfirst

```
3850 \newrobustcmd*{\Glsentryfirst}[1]{%
3851 \protected@edef\@glo@text{\csname glo@#1@first\endcsname}%
3852 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form (as specified by the firstplural key when the entry was defined).

glsentryfirstplural

```
3853 \newcommand*{\glsentryfirstplural}[1]{%
3854 \csname glo@#1@firstpl\endcsname}
```

Glsentryfirstplural

```
3855 \newrobustcmd*{\Glsentryfirstplural}[1]{%
3856 \protected@edef\@glo@text{\csname glo@#1@firstpl\endcsname}%
3857 \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

```
3858 \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

```
3859 \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.

```
3860 \newcommand*{\glsentryuseri}[1]{\csname glo@#1@useri\endcsname}
```

\Glsentryuseri

```
3861 \newrobustcmd*{\Glsentryuseri}[1]{%
3862 \protected@edef\@glo@text{\csname glo@#1@useri\endcsname}%
3863 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glsentryuserii` Get the second user key (as specified by the user2 when the entry was defined).
The argument is the label associated with the entry.

```
3864 \newcommand*{\glsentryuserii}[1]{\csname glo@#1@userii\endcsname}
```

`\Glsentryuserii`

```
3865 \newrobustcmd*{\Glsentryuserii}[1]{%
3866 \protected@edef\@glo@text{\csname glo@#1@userii\endcsname}%
3867 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glsentryuseriii` Get the third user key (as specified by the user3 when the entry was defined).
The argument is the label associated with the entry.

```
3868 \newcommand*{\glsentryuseriii}[1]{\csname glo@#1@useriii\endcsname}
```

`\Glsentryuseriii`

```
3869 \newrobustcmd*{\Glsentryuseriii}[1]{%
3870 \protected@edef\@glo@text{\csname glo@#1@useriii\endcsname}%
3871 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glsentryuseriv` Get the fourth user key (as specified by the user4 when the entry was defined).
The argument is the label associated with the entry.

```
3872 \newcommand*{\glsentryuseriv}[1]{\csname glo@#1@useriv\endcsname}
```

`\Glsentryuseriv`

```
3873 \newrobustcmd*{\Glsentryuseriv}[1]{%
3874 \protected@edef\@glo@text{\csname glo@#1@useriv\endcsname}%
3875 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glsentryuserv` Get the fifth user key (as specified by the user5 when the entry was defined).
The argument is the label associated with the entry.

```
3876 \newcommand*{\glsentryuserv}[1]{\csname glo@#1@userv\endcsname}
```

`\Glsentryuserv`

```
3877 \newrobustcmd*{\Glsentryuserv}[1]{%
3878 \protected@edef\@glo@text{\csname glo@#1@userv\endcsname}%
3879 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glsentryuservi` Get the sixth user key (as specified by the user6 when the entry was defined).
The argument is the label associated with the entry.

```
3880 \newcommand*{\glsentryuservi}[1]{\csname glo@#1@uservi\endcsname}
```

`\Glsentryuservi`

```
3881 \newrobustcmd*{\Glsentryuservi}[1]{%
3882 \protected@edef\@glo@text{\csname glo@#1@uservi\endcsname}%
3883 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glsentryshort` Get the short key (as specified by the short the entry was defined). The argument is the label associated with the entry.

```
3884 \newcommand*{\glsentryshort}[1]{\csname glo@#1@short\endcsname}
```

```

\Glsentryshort
3885 \newrobustcmd*{\Glsentryshort}[1]{%
3886 \protected@edef\@glo@text{\csname glo@#1@short\endcsname}%
3887 \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.
3888 \newcommand*{\glsentryshortpl}[1]{\csname glo@#1@shortpl\endcsname}

\Glsentryshortpl
3889 \newrobustcmd*{\Glsentryshortpl}[1]{%
3890 \protected@edef\@glo@text{\csname glo@#1@shortpl\endcsname}%
3891 \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.
3892 \newcommand*{\glsentrylong}[1]{\csname glo@#1@long\endcsname}

\Glsentrylong
3893 \newrobustcmd*{\Glsentrylong}[1]{%
3894 \protected@edef\@glo@text{\csname glo@#1@long\endcsname}%
3895 \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.
3896 \newcommand*{\glsentrylongpl}[1]{\csname glo@#1@longpl\endcsname}

\Glsentrylongpl
3897 \newrobustcmd*{\Glsentrylongpl}[1]{%
3898 \protected@edef\@glo@text{\csname glo@#1@longpl\endcsname}%
3899 \expandafter\makefirstuc\expandafter{\@glo@text}}

Short cut macros to access full form:

\glsentryfull
3900 \newcommand*{\glsentryfull}[1]{%
3901 \acrfullformat{\glsentrylong{#1}}{\glsentryshort{#1}}%
3902 }

\Glsentryfull
3903 \newrobustcmd*{\Glsentryfull}[1]{%
3904 \acrfullformat{\Glsentrylong{#1}}{\glsentryshort{#1}}%
3905 }

\glsentryfullpl
3906 \newcommand*{\glsentryfullpl}[1]{%
3907 \acrfullformat{\glsentrylongpl{#1}}{\glsentryshortpl{#1}}%
3908 }

```

```

\Glsentryfullpl
3909 \newrobustcmd*{\Glsentryfullpl}[1]{%
3910   \acrfullformat{\Glsentrylongpl{#1}}{\glentryshortpl{#1}}%
3911 }

\glentrynumberlist  Displays the number list as is.
3912 \newcommand*{\glentrynumberlist}[1]{%
3913   \glsoifexists{#1}%
3914   {%
3915     \csname glo@#1@numberlist\endcsname
3916   }%
3917 }

\lsdisplaynumberlist  Formats the number list for the given entry label. Doesn't work with hyperref.
3918 \@ifpackageloaded{hyperref} {%
3919   \newcommand*{\lsdisplaynumberlist}[1]{%
3920     \GlossariesWarning
3921     {%
3922       \string\lsdisplaynumberlist\space
3923       doesn't work with hyperref.^^JUsing
3924       \string\glentrynumberlist\space instead%
3925     }%
3926     \glentrynumberlist{#1}%
3927   }%
3928 }%
3929 {%
3930   \newcommand*{\lsdisplaynumberlist}[1]{%
3931     \glsoifexists{#1}%
3932     {%
3933       \bgroup
3934       \def\@glo@label{#1}%
3935       \let\@org@glnumberformat\glnumberformat
3936       \def\glnumberformat##1{##1}%
3937       \protected@edef\the@numberlist{\csname glo@\@glo@label @numberlist\endcsname}%
3938       \def\@gls@numlist@sep{}%
3939       \def\@gls@numlist@nextsep{}%
3940       \def\@gls@numlist@lastsep{}%
3941       \def\@gls@thislist{}%
3942       \def\@gls@donext@def{}%
3943       \renewcommand\do[1]{%
3944         \protected@edef\@gls@thislist{%
3945           \@gls@thislist
3946           \noexpand\@gls@numlist@sep
3947           ##1%
3948         }%
3949         \let\@gls@numlist@sep\@gls@numlist@nextsep
3950         \def\@gls@numlist@nextsep{\glnumlistsep}%
3951         \@gls@donext@def
3952         \def\@gls@donext@def{%

```



```

3953         \def\@gls@numlist@lastsep{\glsnumlistlastsep}%
3954     }%
3955 }%
3956     \expandafter \glsnumlistparser \expandafter{\the@numberlist}%
3957     \let\@gls@numlist@sep\@gls@numlist@lastsep
3958     \@gls@thislist
3959 \egroup
3960 }%
3961 }
3962 }

```

`\glsnumlistsep`

```

3963 \newcommand*{\glsnumlistsep}{, }

```

`\glsnumlistlastsep`

```

3964 \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.

```

3965 \newcommand*{\glshyperlink}[2][\glsentrytext{\@glo@label}]{%
3966 \def\@glo@label{#2}%
3967 \@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`:

```

3968 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
3969 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}

```

This key is only used by `\glsaddall`:

```

3970 \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`

```

3971 \newrobustcmd*{\glsadd}[2][]{%
3972 \glsdoifexists{#2}%
3973 {%

```

```

3974 \def\@glsnumberformat{glsnumberformat}%
3975 \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
3976 \setkeys{glossadd}{#1}%
Store the entry's counter in \theglsentrycounter
3977 \@gls@saveentrycounter
3978 \@do@wrglossary{#2}%
3979 }%
3980 }

```

\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

```

3981 \newrobustcmd*{\glsaddall}[1] [] {%
3982 \edef\@glo@type{\@glo@types}%
3983 \setkeys{glossadd}{#1}%
3984 \forallglsentries[\@glo@type]{\@glo@entry}{%
3985 \glsadd[#1]{\@glo@entry}%
3986 }%
3987 }

```

\glsaddallunused

\glsaddallunused [*<glossary type>*]

Add all used terms defined for the listed glossaries (without displaying any text). If optional argument is omitted, apply to all glossary types. This should typically go at the end of the document.

```

3988 \newrobustcmd*{\glsaddallunused}[1] [\@glo@types] {%
3989 \forallglsentries[#1]{\@glo@entry}%
3990 {%
3991 \ifglsused{\@glo@entry}{\@glsadd[format=@gobble]{\@glo@entry}}%
3992 }%
3993 }

```

1.12 Creating associated files

The `\writeist` command creates the associated customized `.ist` makeindex style file. While defining this command, some characters have their catcodes temporarily changed to ensure they get written to the `.ist` file correctly. The `makeindex` actual character (usually `@`) is redefined to be a `?`, to allow internal commands to be written to the glossary file output file.

The special characters are stored in `\@gls@actualchar`, `\@gls@encapchar`, `\@gls@levelchar` and `\@gls@quotechar` to make them easier to use later,

but don't change these values, because the characters are encoded in the command definitions that are used to escape the special characters (which means that the user no longer needs to worry about `makeindex` special characters).

The symbols and numbers label for group headings are hardwired into the `.ist` file as `glssymbols` and `glsnumbers`, the group titles can be translated (so that `\glssymbolsgroupname` replaces `glssymbols` and `\glsnumbersgroupname` replaces `glsnumbers`) using the command `\glsgroupname` which is defined in `.ist`. This is done to prevent any problem characters in `\glssymbolsgroupname` and `\glsnumbersgroupname` from breaking hyperlinks.

`\glsopenbrace` Define `\glsopenbrace` to make it easier to write an opening brace to a file.

```
3994 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

`\glsclosebrace` Define `\glsclosebrace` to make it easier to write an opening brace to a file.

```
3995 \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.

```
3996 \edef\glsquote#1{\string"#1\string"}
```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for xindy.

```
3997 \ifglsxindy
```

```
3998   \newcommand*{\@glsfirstletter}{A}
```

```
3999 \fi
```

`\setLetterAfterDigits` Sets the first letter to come after the digits 0,...,9.

```
4000 \ifglsxindy
```

```
4001   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
```

```
4002     \renewcommand*{\@glsfirstletter}{#1}}
```

```
4003 \else
```

```
4004   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
```

```
4005     \glsnoxywarning\GlsSetXdyFirstLetterAfterDigits}
```

```
4006 \fi
```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```
4007 \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.

```
4008 \ifglsxindy
```

```
4009   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
```

```
4010     \renewcommand*{\@glsminrange}{#1}}
```

```
4011 \else
```

```
4012   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
```

```
4013     \glsnoxywarning\GlsSetXdyMinRangeLength}
```

```
4014 \fi
```

```

\writeist
4015 \ifglxindy
    Code to use if xindy is required.
4016 \def\writeist{%
    Update attributes list
4017 \@gls@addpredefinedattributes
    Open the file.
4018 \openout\glswrite=\istfilename
    Write header comment at the start of the file
4019 \write\glswrite{;; xindy style file created by the glossaries
4020 package}%
4021 \write\glswrite{;; for document '\jobname' on
4022 \the\year-\the\month-\the\day}%
    Specify the required styles
4023 \write\glswrite{^^J; required styles^^J}
4024 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4025 \ifx\@xdystyle\@empty
4026 \else
4027 \protected@write\glswrite{{(require
4028 \string"\@xdystyle.xdy\string")}%
4029 \fi
4030 }%
    List the allowed attributes (possible values used by the format key)
4031 \write\glswrite{^^J%
4032 ; list of allowed attributes (number formats)^^J}%
4033 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
    Define any additional alphabets
4034 \write\glswrite{^^J; user defined alphabets^^J}%
4035 \write\glswrite{\@xdyuseralphabets}%
    Define location classes.
4036 \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.
4037 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
    Case were \langle Hprefix\rangle is empty:
4038 \protected@write\glswrite{{(define-location-class
4039 \string"\@gls@classI\string"^^J\space\space\space
4040 (
4041 :sep "{}{"
4042 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4043 :sep "}"
4044 )
4045 ^^J\space\space\space

```

```

4046         :min-range-length \@glsminrange^^J%
4047     )
4048 }%

```

Nested iteration over all classes:

```

4049     {%
4050         \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4051             \protected@write\glswrite{}\{(define-location-class
4052                 \string"\@gls@classII-\@gls@classI\string"
4053                 ^^J\space\space\space
4054                 (
4055                     :sep "{"
4056                     \csname \@gls@xdy@Lclass@\@gls@classII\endcsname\space
4057                     :sep "{{"
4058                     \csname \@gls@xdy@Lclass@\@gls@classI\endcsname\space
4059                     :sep "}"
4060                 )
4061                 ^^J\space\space\space
4062                 :min-range-length \@glsminrange^^J%
4063             )
4064         }%
4065     }%
4066 }%
4067 }%

```

User defined location classes (needs checking for new location format).

```

4068 \write\glswrite{^^J; user defined location classes}%
4069 \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.)

```

4070 \write\glswrite{^^J; define cross-reference class^^J}%
4071 \write\glswrite{(define-crossref-class \string"see\string"
4072     :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.)

```

4073 \write\glswrite{(markup-crossref-list
4074     :class \string"see\string"^^J\space\space\space
4075     :open \string"\string\glsseeformat\string"
4076     :close \string"{}\string")}%

```

List the order to sort the classes.

```

4077 \write\glswrite{^^J; define the order of the location classes}%
4078 \write\glswrite{(define-location-class-order
4079     (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4080 \write\glswrite{^^J; define the glossary markup^^J}%

```

```

4081 \write\glswrite{(markup-index^^J\space\space\space
4082 :open \string"\string
4083 \glossarysection[\string\glossarytoctitle]{\string
4084 \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)

```

4085 \@for\@this@ctr:=\@xdycounters\do{%
4086 {%
4087 \for\@this@attr:=\@xdyattributelist\do{%
4088 \protected\write\glswrite{ }\string\providecommand*%
4089 \expandafter\string
4090 \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4091 {%
4092 \string\setentrycounter
4093 [\expandafter\@gobble\string\#1]{\@this@ctr}%
4094 \expandafter\string
4095 \csname\@this@attr\endcsname
4096 {\expandafter\@gobble\string\#2}%
4097 }%
4098 }%
4099 }%
4100 }%
4101 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

4102 \write\glswrite{%
4103 \string\begin
4104 {theglossary}\string\glossaryheader\string~n\string" ^^J\space
4105 \space\space:close \string"\expandafter\@gobble
4106 \string%\string~n\string
4107 \end{theglossary}\string\glossarypostamble
4108 \string~n\string" ^^J\space\space\space
4109 :tree)}}%

```

Specify what to put between letter groups

```

4110 \write\glswrite{(markup-letter-group-list
4111 :sep \string"\string\glsgrpskip\string~n\string"))%

```

Specify what to put between entries

```

4112 \write\glswrite{(markup-indexentry
4113 :open \string"\string\relax \string\glsresetentrylist
4114 \string~n\string"))%

```

Specify how to format entries

```

4115 \write\glswrite{(markup-locclass-list :open
4116 \string"\glsopenbrace\string\glossaryentrynumbers
4117 \glsopenbrace\string\relax\space \string"^^J\space\space\space
4118 :sep \string", \string"
4119 :close \string"\glsclosebrace\glsresetentrylist\string"))%

```

Specify how to separate location numbers

```
4120 \write\glswrite{(markup-locref-list
4121 :sep \string\string\delimN\space\string)}}%
```

Specify how to indicate location ranges

```
4122 \write\glswrite{(markup-range
4123 :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.

```
4124 \@onelevel@sanitize\gls@suffiF
4125 \@onelevel@sanitize\gls@suffiFF

4126 \ifx\gls@suffiF\@empty
4127 \else
4128 \write\glswrite{(markup-range
4129 :close "\gls@suffiF" :length 1 :ignore-end)}}%
4130 \fi
4131 \ifx\gls@suffiFF\@empty
4132 \else
4133 \write\glswrite{(markup-range
4134 :close "\gls@suffiFF" :length 2 :ignore-end)}}%
4135 \fi
```

Specify how to format locations.

```
4136 \write\glswrite{^^J; define format to use for locations^^J}%
4137 \write\glswrite{\@xdylocref}%
```

Specify how to separate letter groups.

```
4138 \write\glswrite{^^J; define letter group list format^^J}%
4139 \write\glswrite{(markup-letter-group-list
4140 :sep \string\string\glsgroupskip\string~n\string)}}%
```

Define letter group headings.

```
4141 \write\glswrite{^^J; letter group headings^^J}%
4142 \write\glswrite{(markup-letter-group
4143 :open-head \string\string\glsgroupheading
4144 \glsopenbrace\string^^J\space\space\space
4145 :close-head \string\glsclosebrace\string)}}%
```

Define additional letter groups.

```
4146 \write\glswrite{^^J; additional letter groups^^J}%
4147 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
4148 \write\glswrite{^^J; additional sort rules^^J}
4149 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
4150 \closeout\glswrite
```

Suppress any further calls.

```
4151 \let\writeist\relax
4152 }
4153 \else
```

Code to use if makeindex is required.

```
4154 \edef\@gls@actualchar{\string?}
4155 \edef\@gls@encapchar{\string|}
4156 \edef\@gls@levelchar{\string!}
4157 \edef\@gls@quotechar{\string"}
4158 \def\writeist{\relax
4159 \openout\glswrite=\istfilename
4160 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
4161 created by the glossaries package}
4162 \write\glswrite{\expandafter\@gobble\string\% for document
4163 '\jobname' on \the\year-\the\month-\the\day}
4164 \write\glswrite{actual '\@gls@actualchar'}
4165 \write\glswrite{encap '\@gls@encapchar'}
4166 \write\glswrite{level '\@gls@levelchar'}
4167 \write\glswrite{quote '\@gls@quotechar'}
4168 \write\glswrite{keyword \string"\string\glossaryentry\string"}
4169 \write\glswrite{preamble \string"\string\glossarysection[\string
4170 \glossarytoctitle]{\string\glossarytitle}\string
4171 \glossarypreamble\string\n\string\begin{theglossary}\string
4172 \glossaryheader\string\n\string"}
4173 \write\glswrite{postamble \string"\string%\string\n\string
4174 \end{theglossary}\string\glossarypostamble\string\n
4175 \string"}
4176 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
4177 \string"}
4178 \write\glswrite{item_0 \string"\string%\string\n\string"}
4179 \write\glswrite{item_1 \string"\string%\string\n\string"}
4180 \write\glswrite{item_2 \string"\string%\string\n\string"}
4181 \write\glswrite{item_01 \string"\string%\string\n\string"}
4182 \write\glswrite{item_x1
4183 \string"\string\relax \string\glsresetentrylist\string\n
4184 \string"}
4185 \write\glswrite{item_12 \string"\string%\string\n\string"}
4186 \write\glswrite{item_x2
4187 \string"\string\relax \string\glsresetentrylist\string\n
4188 \string"}

4189 \write\glswrite{delim_0 \string"\string\{\string
4190 \glossaryentrynumbers\string\{\string\relax \string"}
4191 \write\glswrite{delim_1 \string"\string\{\string
4192 \glossaryentrynumbers\string\{\string\relax \string"}
4193 \write\glswrite{delim_2 \string"\string\{\string
4194 \glossaryentrynumbers\string\{\string\relax \string"}
4195 \write\glswrite{delim_t \string"\string\}\string\}\string"}
4196 \write\glswrite{delim_n \string"\string\delimN \string"}
```



```

4197 \write\glswrite{delim_r \string"\string\\delimR \string"}
4198 \write\glswrite{headings_flag 1}
4199 \write\glswrite{heading_prefix
4200 \string"\string\\glsgroupheading\string\{\string"}
4201 \write\glswrite{heading_suffix
4202 \string"\string\\}\string\\relax
4203 \string\\glsgresetentrylist \string"}
4204 \write\glswrite{symhead_positive \string"glssymbols\string"}
4205 \write\glswrite{numhead_positive \string"glsgnumbers\string"}
4206 \write\glswrite{page_compositor \string"glsgcompositor\string"}
4207 \@glsgescbsdq\glsgsuffixF
4208 \@glsgescbsdq\glsgsuffixFF
4209 \ifx\glsgsuffixF\@empty
4210 \else
4211 \write\glswrite{suffix_2p \string"\glsgsuffixF\string"}
4212 \fi
4213 \ifx\glsgsuffixFF\@empty
4214 \else
4215 \write\glswrite{suffix_3p \string"\glsgsuffixFF\string"}
4216 \fi
4217 \closeout\glswrite
4218 \let\writeist\relax
4219 }
4220 \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`

```

4221 \newcommand{\noist}{%
Update attributes list
4222 \@glsgaddpredefinedattributes
4223 \let\writeist\relax
4224 }

```

`\@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`

```

4225 \newcommand*{\@makeglossary}[1]{%

```

```

4226 \ifglossaryexists{#1}%
4227 {%
    Only create a new write if savewrites=false otherwise create a token to collect
    the information.
4228 \ifglssavewrites
4229 \expandafter\newtoks\csname glo@#1@filetok\endcsname
4230 \else
4231 \expandafter\newwrite\csname glo@#1@file\endcsname
4232 \expandafter\@glsoopenfile\csname glo@#1@file\endcsname{#1}%
4233 \fi
4234 \@gls@renewglossary
4235 \writeist
4236 }%
4237 {%
4238 \PackageError{glossaries}%
4239 {Glossary type ‘#1’ not defined}%
4240 {New glossaries must be defined before using \string\makeglossary}%
4241 }%
4242 }

```

`\@glsoopenfile` Open write file associated with the given glossary.

```

4243 \newcommand*{\@glsoopenfile}[2]{%
4244 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
4245 \PackageInfo{glossaries}{Writing glossary file
4246 \jobname.\csname @glotype@#2@out\endcsname}%
4247 }

```

`\@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4248 \newcommand*{\@warn@nomakeglossaries}{%
4249 \GlossariesWarningNoLine{\string\makeglossaries\space
4250 hasn't been used,^^Jthe glossaries will not be updated}%
4251 }

```

`\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`

```

4252 \newcommand*{\makeglossaries}{%
    If the user removes the glossary package from their document, ensure the next
    run doesn't throw a load of undefined control sequence errors when the aux file
    is parsed.
4253 \protected@write\@auxout{}{\string\providecommand\string\@glsorder[1]{} }
4254 \protected@write\@auxout{}{\string\providecommand\string\@istfilename[1]{} }
4255 % Write the name of the style file to the aux file
4256 % (needed by \app{makeglossaries})

```

```

4257% \begin{macrocode}
4258 \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
4259 \protected@write\@auxout{}\string\@glsorder{\glsorder}}

Iterate through each glossary type and activate it.
4260 \@for\@glo@type:=\@glo@types\do{%
4261 \ifthenelse{\equal{\@glo@type}{}}{}{}%
4262 \@makeglossary{\@glo@type}}%
4263 }%

New glossaries must be created before \makeglossaries so disable \newglossary.
4264 \renewcommand*\newglossary[4][]{%
4265 \PackageError{glossaries}{New glossaries
4266 must be created before \string\makeglossaries}{You need
4267 to move \string\makeglossaries\space after all your
4268 \string\newglossary\space commands}}%

Any subsequent instances of this command should have no effect
4269 \let\@makeglossary\relax
4270 \let\makeglossary\relax
4271 \let\makeglossaries\relax

Disable all commands that have no effect after \makeglossaries
4272 \@disable@onlypremakeg

Allow see key:
4273 \let\gls@checkseeallowed\relax

Suppress warning about no \makeglossaries
4274 \let\warn@nomakeglossaries\relax

Declare list parser for \glsdisplaynumberlist
4275 \ifglssavenumberlist
4276 \edef\@gls@dodedeflistparser{\noexpand\DeclareListParser
4277 {\noexpand\glsnumlistparser}{\delimN}}%
4278 \@gls@dodedeflistparser
4279 \fi
4280 }

```

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`

```

4281 \let\makeglossary\makeglossaries

If \makeglossaries hasn't been used, issue a warning. Also issue a warning
if neither \printglossaries nor \printglossary have been used.
4282 \AtEndDocument{%
4283 \warn@nomakeglossaries
4284 \warn@noprintglossary
4285 }

```

1.13 Writing information to associated files

`\glswrite` The write used for style file also used for all other output files if `savewrites=true`.

```
4286 \newwrite\glswrite
```

`\istfile` Deprecated.

```
4287 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```
4288 \AtEndDocument{%
```

```
4289   \glswritefiles
```

```
4290 }
```

`\@glswritefiles` Only write the files if `savewrites=true`

```
4291 \newcommand*{\@glswritefiles}{%
```

Iterate through all the glossaries

```
4292   \forallglossaries{\@glo@type}{%
```

Check for empty glossaries (patch provided by Patrick Häcker)

```
4293     \ifcsundef{glo@\@glo@type @filetok}%
```

```
4294     {%
```

```
4295       \def\gls@tmp{}%
```

```
4296     }%
```

```
4297     {%
```

```
4298       \edef\gls@tmp{\expandafter\the
```

```
4299         \csname glo@\@glo@type @filetok\endcsname}%
```

```
4300     }%
```

```
4301     \ifx\gls@tmp\@empty
```

```
4302       \ifx\@glo@type\glsdefaulttype
```

```
4303         \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
4304           entries.^^JRemember to use package option ‘nomain’ if
```

```
4305 you
```

```
4306           don’t want to^^Juse the main glossary}%
```

```
4307       \else
```

```
4308         \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
4309           entries}%
```

```
4310       \fi
```

```
4311     \else
```

```
4312       \@glsopenfile{\glswrite}{\@glo@type}%
```

```
4313       \immediate\write\glswrite{%
```

```
4314         \expandafter\the
```

```
4315         \csname glo@\@glo@type @filetok\endcsname}%
```

```
4316       \immediate\closeout\glswrite
```

```
4317     \fi
```

```
4318   }%
```

```
4319 }
```

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`

```
4320 \renewcommand*{\glossary}[1][\glsdefaultttype]{%
4321   \@glossary[#1]%
4322 }
```

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`

```
4323 \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`

```
4324 \newcommand{\@gls@renewglossary}{%
4325   \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
4326   \let\@gls@renewglossary\empty
4327 }
```

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`

```
4328 \renewcommand*{\@wrglossary}[2]{%
4329   \ifglssavewrites
4330     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
4331     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4332       \expandafter{\@gls@tmp^^J}%
4333   \else
4334     \ifcsdef{glo@#1@file}%
4335       {%
4336         \expandafter\protected@write\csname glo@#1@file\endcsname{%
4337           \gls@disablepagerefexpansion}{#2}%
4338       }%
4339       {%
4340         \GlossariesWarning{No file defined for glossary ‘#1’}%
4341       }%
4342     \fi
4343   \endgroup\@esphack
4344 }
```

```

\do@wrglossary
4345 \newcommand*\do@wrglossary}[1]{%
4346   \ifglindexonlyfirst
4347     \ifglused{#1}{\do@wrglossary{#1}}%
4348   \else
4349     \do@wrglossary{#1}%
4350   \fi
4351 }

```

`@protected@pagefmts` List of page formats to be protected against expansion.

```

4352 \newcommand\gls@protected@pagefmts{%
4353   \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4354 }

```

`blepagerefexpansion`

```

4355 \newcommand*\gls@disablepagerefexpansion{%
4356   \@for\@gls@this:=\gls@protected@pagefmts\do
4357   {%
4358     \expandafter\let\@gls@this\relax
4359   }%
4360 }

```

`\gls@alphpage`

```

4361 \newcommand*\gls@alphpage{\@alph\c@page}

```

`\gls@Alphpage`

```

4362 \newcommand*\gls@Alphpage{\@Alph\c@page}

```

`\gls@numberpage`

```

4363 \newcommand*\gls@numberpage{\number\c@page}

```

`\gls@romanpage`

```

4364 \newcommand*\gls@romanpage{\romannumeral\c@page}

```

`\gls@Romanpage`

```

4365 \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.

```

4366 \newcommand*\do@wrglossary}[1]{%
4367   \begingroup

```

First a bit of hackery to prevent premature expansion of `\c@page`. Store original definitions:

```

4368   \let\orgthe\the
4369   \let\orgnumber\number
4370   \let\orgromannumeral\romannumeral
4371   \let\orgalph\@alph
4372   \let\orgAlph\@Alph
4373   \let\orgRoman\@Roman

```

Redefine:

```
4374 \def\the##1{%
4375 \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4376 \def\number##1{%
4377 \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4378 \def\romannumeral##1{%
4379 \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4380 \def\@Roman##1{%
4381 \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4382 \def\@alph##1{%
4383 \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4384 \def\@Alph##1{%
4385 \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%

```

Prevent expansion:

```
4386 \gls@disablepagerefexpansion

```

Now store location in \@glslocref:

```
4387 \protected@xdef\@glslocref{\theglsentrycounter}%
4388 \endgroup

```

Escape any special characters

```
4389 \@gls@checkmkidxchars\@glslocref

```

Check if the hyper-location is the same as the location and set the hyper prefix.

```
4390 \expandafter\ifx\theHglentrycounter\theglsentrycounter
4391 \def\@glo@counterprefix{%
4392 \else
4393 \protected@edef\@glsHlocref{\theHglentrycounter}%
4394 \@gls@checkmkidxchars\@glsHlocref
4395 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4396 {\@glslocref}{\@glsHlocref}%
4397 }%
4398 \@do@gls@getcounterprefix
4399 \fi

```

Determine whether to use xindy or makeindex syntax

```
4400 \ifglsxindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```
4401 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
4402 \def\@glo@range{%
4403 \expandafter\if\@glo@prefix(\relax
4404 \def\@glo@range{:open-range}%
4405 \else
4406 \expandafter\if\@glo@prefix)\relax
4407 \def\@glo@range{:close-range}%
4408 \fi
4409 \fi

```

Write to the glossary file using xindy syntax.

```

4410 \glossary[\csname glo@#1@type\endcsname]{%
4411 (indexentry :tkey (\csname glo@#1@index\endcsname)
4412 :locoref \string"{\@glo@counterprefix}{\@glslocoref}\string" %
4413 :attr \string"\@gls@counter\@glo@suffix\string"
4414 \@glo@range
4415 )
4416 }%
4417 \else

```

Convert the format information into the format required for makeindex

```

4418 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
4419 {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

4420 \glossary[\csname glo@#1@type\endcsname]{%
4421 \string\glossaryentry{\csname glo@#1@index\endcsname
4422 \@gls@encapchar\@glo@numfmt}{\@glslocoref}}%
4423 \fi
4424 }

```

`\ls@getcounterprefix` Get the prefix that needs to be prepended to counter in order to get the hyper counter. (For example, with the standard article class and hyperref, `\theequation` needs to be prefixed with `\section num`.) NB this assumes that the prefix ends with a dot, which is the standard. (Otherwise it makes the xindy location classes more complicated.)

```

4425 \newcommand*\@gls@getcounterprefix[2]{%
4426 \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4427 \ifx\@gls@thisloc\@gls@thisHloc
4428 \def\@glo@counterprefix{}%
4429 \else
4430 \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4431 \def\@glo@tmp{##2}%
4432 \ifx\@glo@tmp\@empty
4433 \def\@glo@counterprefix{}%
4434 \else
4435 \def\@glo@counterprefix{##1}%
4436 \fi
4437 }%
4438 \@gls@get@counterprefix#2.#1\end@getprefix
4439 \fi
4440 }

```

1.14 Glossary Entry Cross-References

`\@do@seeglossary` Write the glossary entry with a cross reference. The first argument is the entry's label, the second must be in the form `[\langle tag \rangle]{\langle list \rangle}`, where `\langle tag \rangle` is a tag such as "see" and `\langle list \rangle` is a list of labels.


```

4441 \newcommand{\@do@seeglossary}[2]{%
4442 \def\@gls@xref{#2}%
4443 \@onelevel@sanitize\@gls@xref
4444 \@gls@checkmkidxchars\@gls@xref
4445 \ifglsxindy
4446   \glossary[\csname glo@#1@type\endcsname]{%
4447     (indexentry
4448       :tkey (\csname glo@#1@index\endcsname)
4449       :xref (\string"\@gls@xref\string")
4450       :attr \string"see\string"
4451     )
4452   }%
4453 \else
4454   \glossary[\csname glo@#1@type\endcsname]{%
4455     \string\glossaryentry{\csname glo@#1@index\endcsname
4456       \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
4457 \fi
4458 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4459 \def\@gls@fixbraces#1#2#3\@nil{%
4460   \ifx#2[\relax
4461     \def#1{#2#3}%
4462   \else
4463     \def#1{{#2#3}}%
4464   \fi
4465 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

4466 \DeclareRobustCommand*\glssee[3][\seename]{%
4467   \@do@seeglossary{#2}{[#1]{#3}}
4468 \newcommand*\@glssee[3][\seename]{%
4469   \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.

```

4470 \DeclareRobustCommand*\glsseeformat[3][\seename]{%
4471   \emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

4472 \DeclareRobustCommand*\glsseelist[1]{%

```

If there is only one item in the list, set the last separator to do nothing.

```

4473   \let\@gls@dolast\relax

```

Don't display separator on the first iteration of the loop

```

4474   \let\@gls@donext\relax

```

Iterate through the labels

```

4475   \@for\@gls@thislabel:=#1\do{%

```

```

    Check if on last iteration of loop
4476   \ifx\@xfor@nextelement\@nnil
4477     \@gls@dolast
4478   \else
4479     \@gls@donext
4480   \fi

    display the entry for this label
4481   \glsseeitem{\@gls@thislabel}%

    Update separators
4482   \let\@gls@dolast\glsseelastsep
4483   \let\@gls@donext\glsseesep
4484   }%
4485 }

```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
4486 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entries in a cross-referencing list.

```
4487 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4488 \DeclareRobustCommand*{\glsseeitem}[1]{\gls hyperlink[\glsseeitemformat{#1}]{#1}}
```

`\glsseeitemformat` As from v3.0, default is to use `\glsentrytext` instead of `\glsentryname`. (To avoid problems with the name key being sanitized.)

```
4489 \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.

```

4490 \newcommand*{\gls@save@numberlist}[1]{%
4491   \ifglssavenumberlist
4492     \toks@{#1}%
4493     \edef\@do@writeaux@info{%
4494       \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
4495     }%
4496     \@onelevel@sanitize\@do@writeaux@info
4497     \protected@write\@auxout{}\@do@writeaux@info%
4498   \fi
4499 }

```

`\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.)

```
4500 \def\warn@noprintglossary{%
4501   \GlossariesWarningNoLine{No \string\printglossary\space
4502     or \string\printglossaries\space
4503     found.^^JThis document will not have a glossary}%
4504 }
```

`\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.

```
4505 \ifcsundef{printglossary}{}%
4506 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
4507   \GlossariesWarning{Overriding \string\printglossary}%
4508   \undef\printglossary
4509 }
```

`\printglossary` has an optional argument. The default value is to set the glossary
type to the main glossary.

```
4510 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%
```

Set up defaults.

```
4511   \def\@glo@type{\glsdefaulttype}%
4512   \def\glossarytitle{\csname @glotype@\@glo@type @title\endcsname}%

4513   \def\glossarytoctitle{\glossarytitle}%
4514   \let\org@glossarytitle\glossarytitle
4515   \def\@glossarystyle{}%
4516   \def\gls@dotoc@title{\glssettoctitle{\@glo@type}}%
```

Store current value of `\glossaryentrynumbers`. (This may be changed via the
optional argument)

```
4517   \let\@org@glossaryentrynumbers\glossaryentrynumbers
```

Localise the effects of the optional argument

```
4518   \bgroup
```

Determine settings specified in the optional argument.

```
4519   \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)

```
4520   \ifx\glossarytitle\org@glossarytitle
4521   \else
4522     \expandafter\let\csname @glotype@\@glo@type @title\endcsname
4523       \glossarytitle
4524   \fi
```

Allow a high-level user command to indicate the current glossary

```
4525 \let\currentglossary\@glo@type
```

Enable individual number lists to be suppressed.

```
4526 \let\org@glossaryentrynumbers@glossaryentrynumbers
4527 \let\glsnonextpages@glsnonextpages
```

Enable individual number list to be activated:

```
4528 \let\glsnextpages@glsnextpages
```

Enable suppression of description terminators.

```
4529 \let\nopostdesc@nopostdesc
```

Set up the entry for the TOC

```
4530 \gls@dotoc@title
```

Set the glossary style

```
4531 \@glossarystyle
```

added a way to fetch the current entry label (v3.08 updated for new \glossentry and \subglossentry):

```
4532 \let\gls@org@glossaryentryfield@glossentry
4533 \let\gls@org@glossarysubentryfield@subglossentry
4534 \renewcommand{\glossentry}[1]{%
4535   \gdef\glscurrententrylabel{##1}%
4536   \gls@org@glossaryentryfield{##1}%
4537 }%
4538 \renewcommand{\subglossentry}[2]{%
4539   \gdef\glscurrententrylabel{##2}%
4540   \gls@org@glossarysubentryfield{##1}{##2}%
4541 }%
```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter.

```
4542 \makeatletter
```

Input the glossary file, if it exists.

```
4543 \@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.

```
4544 \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
4545 {}%
4546 {\null}%
```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```
4547 \ifglxindy
4548   \ifcsundef{@xdy\@glo@type @language}%
4549   {%
4550     \edef\do@auxoutstuff{%
4551       \noexpand\AtEndDocument{%
```

If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```

4552         \noexpand\immediate\noexpand\write\@auxout{%
4553         \string\providecommand\string\@xdylanguage[2]{}}%
4554         \noexpand\immediate\noexpand\write\@auxout{%
4555         \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4556     }%
4557 }%
4558 }%
4559 {%
4560     \edef\@do@auxoutstuff{%
4561     \noexpand\AtEndDocument{%
4562     \noexpand\immediate\noexpand\write\@auxout{%
4563     \string\providecommand\string\@xdylanguage[2]{}}%
4564     \noexpand\immediate\noexpand\write\@auxout{%
4565     \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4566     @language\endcsname}}%
4567     }%
4568     }%
4569     }%
4570     \@do@auxoutstuff
4571     \edef\@do@auxoutstuff{%
4572     \noexpand\AtEndDocument{%

```

If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```

4573         \noexpand\immediate\noexpand\write\@auxout{%
4574         \string\providecommand\string\@gls@codepage[2]{}}%
4575         \noexpand\immediate\noexpand\write\@auxout{%
4576         \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
4577     }%
4578 }%
4579     \@do@auxoutstuff
4580 \fi
4581 \egroup

```

Reset \glossaryentrynumbers

```

4582 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers

```

Suppress warning about no \printglossary

```

4583 \global\let\warn@noprntglossary\relax
4584 }

```

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`

```
4585 \newcommand*{\printglossaries}{%
4586   \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}}%
4587 }
```

The keys that can be used in the optional argument to `\printglossary` are as follows: The type key sets the glossary type.

```
4588 \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`.

```
4589 \define@key{printgloss}{title}{%
4590   \def\glossarytitle{#1}%
4591   \let\gls@dotoc@title\relax
4592 }
```

The toctitle sets the text used for the relevant entry in the table of contents.

```
4593 \define@key{printgloss}{toctitle}{%
4594   \def\glossarytoctitle{#1}%
4595   \let\gls@dotoc@title\relax
4596 }
```

The style key sets the glossary style (but only for the given glossary).

```
4597 \define@key{printgloss}{style}{%
4598   \ifcsundef{@glsstyle@#1}%
4599   {%
4600     \PackageError{glossaries}%
4601     {Glossary style ‘#1’ undefined}{}%
4602   }%
4603   {%
4604     \def\@glossarystyle{\setglossentrycompatibility
4605       \csname @glsstyle@#1\endcsname}%
4606   }%
4607 }
```

The numberedsection key determines if this glossary should be in a numbered section.

```
4608 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
4609   false,nolabel,autolabel}[nolabel]{%
4610   \ifcase\nr\relax
4611     \renewcommand*{\@glossarysecstar}{*}%
4612     \renewcommand*{\@glossaryseclabel}{}%
4613   \or
4614     \renewcommand*{\@glossarysecstar}{}%
4615     \renewcommand*{\@glossaryseclabel}{}%
4616   }
```

```

4616 \or
4617   \renewcommand*{\@@glossarysecstar}{}%
4618   \renewcommand*{\@@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
4619 \fi}

```

The `nogroupskip` key determines whether or not there should be a vertical gap between glossary groups.

```

4620 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
4621   \csuse{glsnogroupskip#1}%
4622 }

```

The `nonumberlist` key determines if this glossary should have a number list.

```

4623 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
4624   \ifglsnonumberlist
4625     \def\glossaryentrynumbers##1{%
4626   \else
4627     \def\glossaryentrynumbers##1{##1}%
4628 \fi}

```

`\@glsnonextpages` Suppresses the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnonextpages` is place in the entry's description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```

4629 \newcommand*{\@glsnonextpages}{%
4630   \gdef\glossaryentrynumbers##1{%
4631     \glsresetentrylist
4632   }%
4633 }

```

`\@glsnextpages` Activate the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnextpages` is place in the entry's description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```

4634 \newcommand*{\@glsnextpages}{%
4635   \gdef\glossaryentrynumbers##1{%
4636     ##1\glsresetentrylist}}

```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```

4637 \newcommand*{\glsresetentrylist}{%
4638   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```

4639 \newcommand*{\glsnonextpages}{}

```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```

4640 \newcommand*{\glsnextpages}{}

```

glossaryentry If the entrycounter package option has been used, define a counter to number each level 0 entry.

```
4641 \ifglentrycounter
4642   \ifx\@gls@counterwithin\@empty
4643     \newcounter{glossaryentry}
4644   \else
4645     \newcounter{glossaryentry}[\@gls@counterwithin]
4646   \fi
4647   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
4648 \fi
```

glossarysubentry If the subentrycounter package option has been used, define a counter to number each level 1 entry.

```
4649 \ifglssubentrycounter
4650   \ifglentrycounter
4651     \newcounter{glossarysubentry}[glossaryentry]
4652   \else
4653     \newcounter{glossarysubentry}
4654   \fi
4655   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
4656 \fi
```

resetsubentrycounter Resets the glossarysubentry counter.

```
4657 \ifglssubentrycounter
4658   \newcommand*\glsresetsubentrycounter{%
4659     \setcounter{glossarysubentry}{0}%
4660   }
4661 \else
4662   \newcommand*\glsresetsubentrycounter{}
4663 \fi
```

resetentrycounter Resets the glossaryentry counter.

```
4664 \ifglentrycounter
4665   \newcommand*\glsresetentrycounter{%
4666     \setcounter{glossaryentry}{0}%
4667   }
4668 \else
4669   \newcommand*\glsresetentrycounter{}
4670 \fi
```

\glsstepentry Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```
4671 \ifglentrycounter
4672   \newcommand*\glsstepentry[1]{%
4673     \refstepcounter{glossaryentry}%
4674     \label{glsentry-#1}%
4675   }
4676 \else
```



```

4677 \newcommand*{\glsstepentry}[1]{%
4678 \fi

```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

4679 \ifglssubentrycounter
4680 \newcommand*{\glsstepsubentry}[1]{%
4681 \def\currentglssubentry{#1}%
4682 \refstepcounter{glossarysubentry}%
4683 \label{glsentry-#1}%
4684 }
4685 \else
4686 \newcommand*{\glsstepsubentry}[1]{%
4687 \fi

```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

4688 \ifglssentrycounter
4689 \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
4690 \else
4691 \ifglssubentrycounter
4692 \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
4693 \else
4694 \newcommand*{\glsrefentry}[1]{\gls{#1}}
4695 \fi
4696 \fi

```

`glsentrycounterlabel` Defines how to display the glossaryentry counter.

```

4697 \ifglssentrycounter
4698 \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
4699 \else
4700 \newcommand*{\glsentrycounterlabel}{}
4701 \fi

```

`glssubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

4702 \ifglssubentrycounter
4703 \newcommand*{\glssubentrycounterlabel}{\theglossarysubentry)\space}
4704 \else
4705 \newcommand*{\glssubentrycounterlabel}{}
4706 \fi

```

`\glsentryitem` Step and display glossaryentry counter, if appropriate.

```

4707 \ifglssentrycounter
4708 \newcommand*{\glsentryitem}[1]{%
4709 \glsstepentry{#1}\glsentrycounterlabel
4710 }
4711 \else
4712 \newcommand*{\glsentryitem}[1]{\glsresetsubentrycounter}
4713 \fi

```

`\glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```
4714 \ifglssubentrycounter
4715   \newcommand*{\glssubentryitem}[1]{%
4716     \glstepsubentry{#1}\glssubentrycounterlabel
4717   }
4718 \else
4719   \newcommand*{\glssubentryitem}[1]{%
4720 \fi
```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```
4721 \ifcsundef{theglossary}%
4722 {%
4723   \newenvironment{theglossary}{}{}%
4724 }%
4725 {%
4726   \GlossariesWarning{overriding ‘theglossary’ environment}%
4727   \renewenvironment{theglossary}{}{}%
4728 }
```

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`

```
4729 \newcommand*{\glossaryheader}{}%
```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```
4730 \newcommand*{\glstarget}[2]{\@glstarget{\glolinkprefix#1}{#2}}
```

As from version 3.08, glossary information is now written to the external files using `\glossentry` and `\subglossentry` instead of `\glossaryentryfield` and `\glossarysubentryfield`. The default definition provides backward compatibility for glossary styles that use the old forms.

`compatibleglossentry`

`\glossentry{<label>}{<page-list>}`

```
4731 \providecommand*{\compatibleglossentry}[2]{%
4732   \toks@{#2}%
4733   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
```

```

4734     {\noexpand\glsnamefont
4735       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
4736     {\expandafter\expandonce\csname glo@#1@desc\endcsname}}}%
4737     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}}}%
4738     {\the\toks@}%
4739   }%
4740   \@do@glossentry
4741 }

```

\glossentryname

```

4742 \newcommand*{\glossentryname}[1]{%
4743   \glsdoifexists{#1}%
4744   {%
4745     \letcs{\glo@name}{glo@#1@name}%
4746     \expandafter\glsnamefont\expandafter{\glo@name}%
4747   }%
4748 }

```

\Glossentryname

```

4749 \newcommand*{\Glossentryname}[1]{%
4750   \glsdoifexists{#1}%
4751   {%
4752     \glsnamefont{\Glsentryname{#1}}}%
4753   }%
4754 }

```

\glossentrydesc

```

4755 \newcommand*{\glossentrydesc}[1]{%
4756   \glsdoifexists{#1}%
4757   {%
4758     \glsentrydesc{#1}%
4759   }%
4760 }

```

\Glossentrydesc

```

4761 \newcommand*{\Glossentrydesc}[1]{%
4762   \glsdoifexists{#1}%
4763   {%
4764     \Glsentrydesc{#1}%
4765   }%
4766 }

```

\glossentrysymbol

```

4767 \newcommand*{\glossentrysymbol}[1]{%
4768   \glsdoifexists{#1}%
4769   {%
4770     \glsentrysymbol{#1}%
4771   }%
4772 }

```

`\Glossentrysymbol`

```
4773 \newcommand*\Glossentrysymbol}[1]{%
4774   \glsdoifexists{#1}%
4775   {%
4776     \Glsentrysymbol{#1}%
4777   }%
4778 }
```

`\compatiblesubglossentry` `\subglossentry{<level>}{<label>}{<page-list>}`

```
4779 \providecommand*\compatiblesubglossentry}[3]{%
4780   \toks@{#3}%
4781   \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
4782     {#2}%
4783     {\noexpand\glsnamefont
4784       {\expandafter\expandonce\csname glo@#2@name\endcsname}}}%
4785   {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
4786   {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
4787   {\the\toks@}%
4788   }%
4789   \@do@subglossentry
4790 }
```

`\sentrycompatibility`

```
4791 \newcommand*\setglossentrycompatibility{%
4792   \let\glossentry\compatibleglossentry
4793   \let\subglossentry\compatiblesubglossentry
4794 }
4795 \setglossentrycompatibility
```

`\glossaryentryfield` `\glossaryentryfield{<label>}{<name>}{<description>}{<symbol>}{<page-list>}`

This command formerly governed how each entry row should be formatted in the glossary. Now deprecated.

```
4796 \newcommand*\glossaryentryfield}[5]{%
4797   \GlossariesWarning
4798   {Deprecated use of \string\glossaryentryfield.^^J
4799     I recommend you change to \string\glossentry.^^J
4800     If you've just upgraded, try removing your gls auxiliary
4801     files^^J and recompile}%
4802   \noindent\textbf{\glstarget{#1}{#2}} #4 #3. #5\par}
```

`\glossarysubentryfield` `\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 $\langle symbol \rangle$. The first argument is a number indicating the level. (The level should be greater than or equal to 1.)

```
4803 \newcommand*{\glossarysubentryfield}[6]{%
4804   \GlossariesWarning
4805   {Deprecated use of \string\glossarysubentryfield.^^J
4806    I recommend you change to \string\subglossentry.^^J
4807    If you've just upgraded, try removing your gls auxiliary
4808    files^^J and recompile}%
4809   \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`

```
4810 \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`

```
4811 \newcommand*{\glsgroupheading}[1]{}
```

It is possible to “trick” `makeindex` into treating entries as though they belong to the same group, even if the terms don’t start with the same letter, by modifying the sort key. For example, all entries belonging to one group could be defined so that the sort key starts with an a, while entries belonging to another group could be defined so that the sort key starts with a b, and so on. If you want each group to have a heading, you would then need to modify the translation control sequences `\glsgetgrouptitle` and `\glsgetgrouplabel` so that the label is translated into the required title (and vice-versa).

`\glsgetgrouptitle{\langle label \rangle}`

This command produces the title for the glossary group whose label is given by $\langle label \rangle$. 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. Languages other than English may produce labels that are non-expandable, so we need to check for that otherwise it will create a “missing `\endcsname` inserted” error.

`\glsgetgrouptitle`

```
4812 \newcommand*\glsgetgrouptitle}[1]{%
4813   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
4814   \@gls@grptitle
4815 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
4816 \newcommand*\@gls@getgrouptitle}[2]{%
```

Even if the argument appears to be a single letter, it won't be considered a single letter by `\dtl@ifsingle` if it's an active character.

```
4817 \dtl@ifsingle{#1}%
4818 {%
4819   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
4820 }%
4821 {%
4822   \ifboolexpr{test{\ifstrequal{#1}{glssymbols}}
4823               or test{\ifstrequal{#1}{glsnumbers}}}%
4824   {%
4825     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
4826   }%
4827   {%
4828     \def#2{#1}%
4829   }%
4830 }%
4831 }
```

`\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`

```
4832 \newcommand*\glsgetgrouplabel}[1]{%
4833 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
4834 \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`

```
4835 \newcommand*{\setentrycounter}[2][1]{%
4836   \def\@glo@counterprefix{#1}%
4837   \ifx\@glo@counterprefix\@empty
4838     \def\@glo@counterprefix{.}%
4839   \else
4840     \def\@glo@counterprefix{.#1.}%
4841   \fi
4842   \def\glsetentrycounter{#2}%
4843 }
```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```
4844 \newcommand*{\setglossarystyle}[1]{%
4845   \ifcsundef{@glsstyle@#1}%
4846   {%
4847     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
4848   }%
4849   {%
4850     \csname @glsstyle@#1\endcsname
4851   }%
4852 }
```

`\glossarystyle`

```
4853 \newcommand*{\glossarystyle}[1]{%
4854   \ifcsundef{@glsstyle@#1}%
4855   {%
4856     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
4857   }%
4858   {%
4859     \GlossariesWarning
4860     {Deprecated command \string\glossarystyle.^~J
4861     I recommend you switch to \string\setglossarystyle\space unless
4862     you want to maintain backward compatibility}%
4863     \setglossentrycompatibility
4864     \csname @glsstyle@#1\endcsname

4865     \ifcsdef{@glscompstyle@#1}%
4866     {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
4867     {}%
4868   }%
4869 }
```

`\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.

```

4870 \newcommand{\newglossarystyle}[2]{%
4871   \ifcsundef{@glsstyle@#1}%
4872   {%
4873     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
4874   }%
4875   {%
4876     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
4877   }%
4878 }

```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```

4879 \newcommand{\renewglossarystyle}[2]{%
4880   \ifcsundef{@glsstyle@#1}%
4881   {%
4882     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
4883   }%
4884   {%
4885     \csdef{@glsstyle@#1}{#2}%
4886   }%
4887 }

```

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`

```

4888 \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`

```
4889 \ifcsundef{hyperlink}%
4890 {%
4891   \def\glshypernumber#1{#1}%
4892 }%
4893 {%
4894   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
4895 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
4896 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
4897   \ifx\#1\%
4898   \else
4899     \@delimR#1\delimR\delimR\%
4900   \fi
4901   \ifx\#2\%
4902   \else
4903     #2%
4904   \fi
4905   \ifx\#3\%
4906   \else
4907     \@glshypernumber#3\@nil
4908   \fi
4909 }
```

`\@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`

```
4910 \def\@delimR#1\delimR #2\delimR #3\%
4911 \ifx\#2\%
4912   \@delimN{#1}%
4913 \else
4914   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
4915 \fi}
```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```
4916 \def\@delimN#1{\@delimN#1\delimN \delimN\%
4917 \def\@delimN#1\delimN #2\delimN#3\%
4918 \ifx\#3\%
4919   \@gls@numberlink{#1}%
4920 \else
4921   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
4922 }
```

```
4922 \fi
4923 }
```

The following code is modified from hyperref's \HyInd@pagelink where the name of the counter being used is given by \@gls@counter.

```
4924 \def\@gls@numberlink#1{%
4925 \begingroup
4926 \toks@={}%
4927 \@gls@removespaces#1 \@nil
4928 \endgroup}

4929 \def\@gls@removespaces#1 #2\@nil{%
4930 \toks@=\expandafter{\the\toks@#1}%
4931 \ifx\#2\%
4932 \edef\x{\the\toks@}%
4933 \ifx\x\empty
4934 \else

4935 \hyperlink{\glsentrycounter\@glo@counterprefix\the\toks@}%
4936 {\the\toks@}%
4937 \fi
4938 \else
4939 \@gls@ReturnAfterFi{%
4940 \@gls@removespaces#2\@nil
4941 }%
4942 \fi
4943 }
4944 \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

```
4945 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

\hypersf

```
4946 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

\hypertt

```
4947 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

\hyperbf

```
4948 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}
```

\hypermd

```
4949 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}
```

\hyperit

```
4950 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}
```

```

\hypersl
4951 \newcommand*{\hypersl}[1]{\textsl{\glshypernumber{#1}}}

\hyperup
4952 \newcommand*{\hyperup}[1]{\textup{\glshypernumber{#1}}}

\hypersc
4953 \newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}

\hyperemph
4954 \newcommand*{\hyperemph}[1]{\emph{\glshypernumber{#1}}}

```

1.16 Acronyms

```
\oldacronym \oldacronym[⟨label⟩]{⟨abbrv⟩}{⟨long⟩}{⟨key-val list⟩}
```

This emulates the way the old package defined acronyms. It is equivalent to `\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrv⟩}{⟨long⟩}` and it additionally defines the command `\⟨label⟩` which is equivalent to `\gls{⟨label⟩}` (thus `⟨label⟩` must only contain alphabetical characters). If `⟨label⟩` is omitted, `⟨abbrv⟩` is used. This only emulates the syntax of the old package. The way the acronyms appear in the list of acronyms is determined by the definition of `\newacronym` and the glossary style.

Note that `\⟨label⟩` can't have an optional argument if the package is loaded. If hasn't been loaded then you can do `\⟨label⟩[⟨insert⟩]` but you can't do `\⟨label⟩[⟨key-val list⟩]`. For example if you define the acronym `svm`, then you can do `\svm['s]` but you can't do `\svm[format=textbf]`. If the package is loaded, `\svm['s]` will appear as `svm ['s]` which is unlikely to be the desired result. In this case, you will need to use `\gls` explicitly, e.g. `\gls{svm}['s]`.

Note that it is up to the user to load if desired.

```

4955 \newcommand{\oldacronym}[4][\gls@label]{%
4956   \def\gls@label{#2}%
4957   \newacronym[#4]{#1}{#2}{#3}%
4958   \ifcsundef{xspace}%
4959   {%
4960     \expandafter\edef\csname#1\endcsname{%
4961       \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}%
4962     }%
4963   }%
4964   {%
4965     \expandafter\edef\csname#1\endcsname{%
4966       \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
4967         \noexpand\gls{#1}\noexpand\xspace}%
4968     }%
4969   }%
4970 }

```

`\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`

```
4971 \newcommand{\newacronym}[4] [] {}
```

Set up some convenient short cuts. These need to be changed if `\newacronym` is changed (or if the description key is changed).

`\acrpluralsuffix` Plural suffix used by `\newacronym`. This just defaults to `\glspluralsuffix` but is changed to include `\textup` if the `smallcaps` option is used, so that the suffix doesn't appear in small caps as it doesn't look right. For example, `ABCS` looks as though the “s” is part of the acronym, but `ABCS` looks as though the “s” is a plural suffix. Since the entire text `abcs` is set in `\textsc`, `\textup` is needed to cancel it out.

```
4972 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```
4973 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
4974 \newcommand*{\glsshortkey}{short}
```

`\glsshortpluralkey`

```
4975 \newcommand*{\glsshortpluralkey}{shortplural}
```

`\glslongkey`

```
4976 \newcommand*{\glslongkey}{long}
```

`\glslongpluralkey`

```
4977 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
4978 \newrobustcmd*{\acrfull}{%
4979 \@ifstar\s@acrfull\ns@acrfull
4980 }
```

```

4981 \newcommand*\s@acrfull[2][{}]{%
4982   \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}{%
4983     {\@acrfull{hyper=false,#1}{#2}[]}%
4984 }
4985 \newcommand*\ns@acrfull[2][{}]{%
4986   \new@ifnextchar[{\@acrfull{#1}{#2}}{%
4987     {\@acrfull{#1}{#2}[]}%
4988 }

```

Low-level macro:

```

4989 \def\@acrfull#1#2[#3]{%
4990   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
4991 }

```

`\acrlinkfullformat` Format for full links like `\acrfull`. Syntax: `\acrlinkfullformat{<long cs>}{<short cs>}{<options>}{<label>}{<insert>}`

```

4992 \newcommand{\acrlinkfullformat}[5]{%
4993   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}%
4994 }

```

`\acrfullformat` Default full form is `<long>` (`<short>`).

```

4995 \newcommand{\acrfullformat}[2]{#1\space(#2)}

```

Default format for full acronym

`\Acrfull`

```

4996 \newrobustcmd*\Acrfull{%
4997   \@ifstar\s@Acrfull\ns@Acrfull
4998 }

4999 \newcommand*\s@Acrfull[2][{}]{%
5000   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}{%
5001     {\@Acrfull{hyper=false,#1}{#2}[]}%
5002 }
5003 \newcommand*\ns@Acrfull[2][{}]{%
5004   \new@ifnextchar[{\@Acrfull{#1}{#2}}{%
5005     {\@Acrfull{#1}{#2}[]}%
5006 }

```

Low-level macro:

```

5007 \def\@Acrfull#1#2[#3]{%
5008   \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
5009 }

```

`\ACRfull`

```

5010 \newrobustcmd*\ACRfull{%
5011   \@ifstar\s@ACRfull\ns@ACRfull
5012 }

```

```

5013 \newcommand*\s@ACRfull[2][]{%
5014   \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}{%
5015     {\@ACRfull{hyper=false,#1}{#2}[]}%
5016 }
5017 \newcommand*\ns@ACRfull[2][]{%
5018   \new@ifnextchar[{\@ACRfull{#1}{#2}}{%
5019     {\@ACRfull{#1}{#2}[]}%
5020 }

```

Low-level macro:

```

5021 \def\@ACRfull#1#2[#3]{%
5022   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
5023 }

```

Plural:

\acrfullpl

```

5024 \newrobustcmd*\acrfullpl{%
5025   \@ifstar\s@acrfullpl\ns@acrfullpl
5026 }

5027 \newcommand*\s@acrfullpl[2][]{%
5028   \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}{%
5029     {\@acrfullpl{hyper=false,#1}{#2}[]}%
5030 }
5031 \newcommand*\ns@acrfullpl[2][]{%
5032   \new@ifnextchar[{\@acrfullpl{#1}{#2}}{%
5033     {\@acrfullpl{#1}{#2}[]}%
5034 }

```

Low-level macro:

```

5035 \def\@acrfullpl#1#2[#3]{%
5036   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5037 }

```

\Acrfullpl

```

5038 \newrobustcmd*\Acrfullpl{%
5039   \@ifstar\s@Acrfullpl\ns@Acrfullpl
5040 }

5041 \newcommand*\s@Acrfullpl[2][]{%
5042   \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}{%
5043     {\@Acrfullpl{hyper=false,#1}{#2}[]}%
5044 }
5045 \newcommand*\ns@Acrfullpl[2][]{%
5046   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}{%
5047     {\@Acrfullpl{#1}{#2}[]}%
5048 }

```

Low-level macro:

```
5049 \def\@Acrfullpl#1#2[#3]{%
5050   \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5051 }
```

\ACRfullpl

```
5052 \newrobustcmd*{\ACRfullpl}{%
5053   \@ifstar\s@ACRfullpl\ns@ACRfullpl
5054 }

5055 \newcommand*\s@ACRfullpl[2][]{%
5056   \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
5057   {\@ACRfullpl{hyper=false,#1}{#2}[]}%
5058 }
5059 \newcommand*\ns@ACRfullpl[2][]{%
5060   \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%
5061   {\@ACRfullpl{#1}{#2}[]}%
5062 }
```

Low-level macro:

```
5063 \def\@ACRfullpl#1#2[#3]{%
5064   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
5065 }
```

1.17 Predefined acronym styles

\acronymfont This is only used with the additional acronym styles:

```
5066 \newcommand{\acronymfont}[1]{#1}
```

\firstacronymfont This is only used with the additional acronym styles:

```
5067 \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.

```
5068 \newcommand*\acrnameformat[2]{\acronymfont{#1}}
```

Define some tokens used by \newacronym:

\glskeylisttok

```
5069 \newtoks\glskeylisttok
```

\glslabeltok

```
5070 \newtoks\glslabeltok
```

\glsshorttok

```
5071 \newtoks\glsshorttok
```

\glslongtok

```
5072 \newtoks\glslongtok
```

`\newacronymhook` Provide a hook for `\newacronym`:

```

5073 \newcommand*{\newacronymhook}{%

```

`AcronymDisplayStyle` Sets the default acronym display style for given glossary.

```

5074 \newcommand*{\SetDefaultAcronymDisplayStyle}[1]{%
5075   \def\glsentryfmt[#1]{\glsentryfmt}%
5076 }

```

`defaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```

5077 \newcommand*{\DefaultNewAcronymDef}{%
5078   \edef\@do@newglossaryentry{%
5079     \noexpand\newglossaryentry{\the\glslabeltok}%
5080     {%
5081       type=\acronymtype,%
5082       name={\the\glsshorttok},%
5083       sort={\the\glsshorttok},%
5084       text={\the\glsshorttok},%
5085       first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
5086       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5087       firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%
5088                   {\noexpand\expandonce\noexpand\@glo@shortpl}},%
5089       short={\the\glsshorttok},%
5090       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5091       long={\the\glslongtok},%
5092       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5093       description={\the\glslongtok},%
5094       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

Remaining options specified by the user:

```

5095       \the\glskeylisttok
5096     }%
5097   }%
5098   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5099   \let\@org@gls@assign@plural\gls@assign@plural
5100   \let\@org@gls@assign@descplural\gls@assign@descplural
5101   \def\gls@assign@firstpl##1##2{%
5102     \@@gls@expand@field{##1}{firstpl}{##2}%
5103   }%
5104   \def\gls@assign@plural##1##2{%
5105     \@@gls@expand@field{##1}{plural}{##2}%
5106   }%
5107   \def\gls@assign@descplural##1##2{%
5108     \@@gls@expand@field{##1}{descplural}{##2}%
5109   }%
5110   \@do@newglossaryentry
5111   \let\gls@assign@firstpl\@org@gls@assign@firstpl
5112   \let\gls@assign@plural\@org@gls@assign@plural

```



```

5113 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5114 }

```

DefaultAcronymStyle Set up the default acronym style:

```

5115 \newcommand*\SetDefaultAcronymStyle{%
    Set the display style:
5116 \@for\@gls@type:=\@glsacronymlists\do{%
5117 \SetDefaultAcronymDisplayStyle{\@gls@type}%
5118 }%

```

Set up the definition of \newacronym:

```

5119 \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).

```

5120 \ifx\@glsacronymlists\@empty
5121 \def\@glo@type{\acronymtype}%
5122 \setkeys{glossentry}{##1}%
5123 \DeclareAcronymList{\@glo@type}%
5124 \SetDefaultAcronymDisplayStyle{\@glo@type}%
5125 \fi
5126 \glskeylisttok{##1}%
5127 \glslabeltok{##2}%
5128 \glsshorttok{##3}%
5129 \glslongtok{##4}%
5130 \newacronymhook
5131 \DefaultNewAcronymDef
5132 }%
5133 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
5134 }

```

\acrfootnote Used by the footnote acronym styles.

```

5135 \newcommand*\acrfootnote[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

\acrlinkfootnote

```

5136 \newcommand*\acrlinkfootnote[3]{%
5137 \footnote{\glslink{#1}{#2}{#3}}%
5138 }

```

\acrlinkfootnote

```

5139 \newcommand*\acrlinkfootnote[3]{%
5140 \footnote{#3}%
5141 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary for the description and footnote combination.

```

5142 \newcommand*\SetDescriptionFootnoteAcronymDisplayStyle[1]{%

```

```

5143 \defglentryfmt[#1]{%
5144   \ifglused{\glslabel}%
5145   {%
5146     \acronymfont{\glsgenentryfmt}%
5147   }%
5148   {%
5149     \firstacronymfont{\glsgenentryfmt}%
5150     \ifglshassymbol{\glslabel}%
5151     {%
5152       \expandafter\protect\expandafter\acrfootnote\expandafter
5153       {\@gls@link@opts}{\@gls@link@label}%
5154     }%
5155     \glsifplural
5156     {\glsentrysymbolplural{\glslabel}}%
5157     {\glsentrysymbol{\glslabel}}%
5158   }%
5159 }%
5160 }%
5161 }%
5162 }

```

otnoteNewAcronymDef

```

5163 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
5164   \edef\@do@newglossaryentry{%
5165     \noexpand\newglossaryentry{\the\glslabeltok}%
5166     {%
5167       type=\acronymtype,%
5168       name={\noexpand\acronymfont{\the\glsshorttok}},%
5169       sort={\the\glsshorttok},%
5170       first={\the\glsshorttok},%
5171       firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5172       text={\the\glsshorttok},%
5173       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5174       short={\the\glsshorttok},%
5175       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5176       long={\the\glslongtok},%
5177       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5178       symbol={\the\glslongtok},%
5179       symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5180       \the\glskeylisttok
5181     }%
5182   }%
5183   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5184   \let\@org@gls@assign@plural\gls@assign@plural
5185   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5186   \def\gls@assign@firstpl##1##2{%
5187     \@@gls@expand@field{##1}{firstpl}{##2}%
5188   }%
5189   \def\gls@assign@plural##1##2{%

```

```

5190 \@@gls@expand@field{##1}{plural}{##2}%
5191 }%
5192 \def\gls@assign@symbolplural##1##2{%
5193 \@@gls@expand@field{##1}{symbolplural}{##2}%
5194 }%
5195 \do@newglossaryentry
5196 \let\gls@assign@plural\@org@gls@assign@plural
5197 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5198 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5199 }

```

footnoteAcronymStyle If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

5200 \newcommand*\SetDescriptionFootnoteAcronymStyle{%
5201 \renewcommand{\newacronym}[4][\]{%
5202 \ifx\@glsacronymlists\@empty
5203 \def\@glo@type{\acronymtype}%
5204 \setkeys{glossentry}{##1}%
5205 \DeclareAcronymList{\@glo@type}%
5206 \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
5207 \fi
5208 \glskeylisttok{##1}%
5209 \glslabeltok{##2}%
5210 \glsshorttok{##3}%
5211 \glslongtok{##4}%
5212 \newacronymhook
5213 \DescriptionFootnoteNewAcronymDef
5214 }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

5215 \@for\@gls@type:=\@glsacronymlists\do{%
5216 \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
5217 }%

```

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.

```

5218 \ifglsacrsmallcaps
5219 \renewcommand*\acronymfont{[1]{\textsc{##1}}}%
5220 \renewcommand*\acrpluralsuffix{%
5221 \glstextup{\glspluralsuffix}}%
5222 \else
5223 \ifglsacrsmaller
5224 \renewcommand*\acronymfont{[1]{\textsmaller{##1}}}%
5225 \fi
5226 \fi

```

Check for package option clash

```

5227 \ifglsacrdua
5228 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
5229 can’t both be set}{}%
5230 \fi
5231 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with description and dua combination.

```

5232 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
5233 \defglsentryfmt[#1]{\glsenentryfmt}%
5234 }

```

ionDUANewAcronymDef

```

5235 \newcommand*{\DescriptionDUANewAcronymDef}{%
5236 \edef\@do@newglossaryentry{%
5237 \noexpand\newglossaryentry{\the\glslabeltok}%
5238 {%
5239 type=\acronymtype,%
5240 name={\the\glslongtok},%
5241 sort={\the\glslongtok},
5242 text={\the\glslongtok},%
5243 first={\the\glslongtok},%
5244 plural={\noexpand\expandonce\noexpand\@glo@longpl},%
5245 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5246 short={\the\glsshorttok},%
5247 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5248 long={\the\glslongtok},%
5249 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5250 symbol={\the\glsshorttok},%
5251 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5252 \the\glskeylisttok
5253 }%
5254 }%
5255 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5256 \let\@org@gls@assign@plural\gls@assign@plural
5257 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5258 \def\gls@assign@firstpl##1##2{%
5259 \@@gls@expand@field{##1}{firstpl}{##2}%
5260 }%
5261 \def\gls@assign@plural##1##2{%
5262 \@@gls@expand@field{##1}{plural}{##2}%
5263 }%
5264 \def\gls@assign@symbolplural##1##2{%
5265 \@@gls@expand@field{##1}{symbolplural}{##2}%
5266 }%
5267 \do@newglossaryentry
5268 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5269 \let\gls@assign@plural\@org@gls@assign@plural

```

```

5270 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5271 }

```

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.

```

5272 \newcommand*\SetDescriptionDUAAcronymStyle{%
5273   \ifglsacrsmallcaps
5274     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
5275       can't both be set}{}%
5276   \else
5277     \ifglsacrsmaller
5278       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
5279         can't both be set}{}%
5280   \fi
5281 \fi
5282 \renewcommand*\newacronym[4][[]]{%
5283   \ifx\@glsacronymlists\@empty
5284     \def\@glo@type{\acronymtype}%
5285     \setkeys{glossentry}{##1}%
5286     \DeclareAcronymList{\@glo@type}%
5287     \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
5288   \fi
5289   \glskeylisttok{##1}%
5290   \glslabeltok{##2}%
5291   \glsshorttok{##3}%
5292   \glslongtok{##4}%
5293   \newacronymhook
5294   \DescriptionDUANewAcronymDef
5295 }%

```

Set display.

```

5296 \@for\@gls@type:=\@glsacronymlists\do{%
5297   \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
5298 }%
5299 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

5300 \newcommand*\SetDescriptionAcronymDisplayStyle[1]{%
5301   \defglsentryfmt[1]{%
5302     \ifglsused{\glslabel}%
5303     {%

```

Move the inserted text outside of \acronymfont

```

5304     \let\gls@org@insert\glsinsert
5305     \let\glsinsert\@empty
5306     \acronymfont{\glsentryfmt}\gls@org@insert
5307   }%
5308   {%

```

```

5309 \glsgenentryfmt
5310 \ifglshassymbol{\glslabel}%
5311 {%
5312 \glsifplural
5313 {%
5314 \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
5315 }%
5316 {%
5317 \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
5318 }%
5319 \space(\protect\firstacronymfont
5320 {\glscapscase
5321 {\@glo@symbol}
5322 {\@glo@symbol}
5323 {\mfirstucMakeUppercase{\@glo@symbol}}})%
5324 }%
5325 {}%
5326 }%
5327 }%
5328 }

```

ptionNewAcronymDef

```

5329 \newcommand*{\DescriptionNewAcronymDef}{%
5330 \edef\@do@newglossaryentry{%
5331 \noexpand\newglossaryentry{\the\glslabeltok}%
5332 {%
5333 type=\acronymtype,%
5334 name={\noexpand
5335 \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
5336 sort={\the\glsshorttok},%
5337 first={\the\glslongtok},%
5338 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5339 text={\the\glsshorttok},%
5340 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5341 short={\the\glsshorttok},%
5342 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5343 long={\the\glslongtok},%
5344 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5345 symbol={\noexpand\@glo@text},%
5346 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5347 \the\glskeylisttok}%
5348 }%
5349 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5350 \let\@org@gls@assign@plural\gls@assign@plural
5351 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5352 \def\gls@assign@firstpl##1##2{%
5353 \@@gls@expand@field{##1}{firstpl}{##2}%
5354 }%
5355 \def\gls@assign@plural##1##2{%

```

```

5356 \@@gls@expand@field{##1}{plural}{##2}%
5357 }%
5358 \def\gls@assign@symbolplural##1##2{%
5359 \@@gls@expand@field{##1}{symbolplural}{##2}%
5360 }%
5361 \do@newglossaryentry
5362 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5363 \let\gls@assign@plural\@org@gls@assign@plural
5364 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5365 }

```

DescriptionAcronymStyle Option description is used, but not dua or footnote. Store long form in first key and short form in text and symbol key. The name is stored using `\acrnameformat` to allow the user to override the way the name is displayed in the list of acronyms.

```

5366 \newcommand*{\SetDescriptionAcronymStyle}{%
5367 \renewcommand{\newacronym}[4][[]]{%
5368 \ifx\@glsacronymlists\@empty
5369 \def\@glo@type{\acronymtype}%
5370 \setkeys{glossentry}{##1}%
5371 \DeclareAcronymList{\@glo@type}%
5372 \SetDescriptionAcronymDisplayStyle{\@glo@type}%
5373 \fi
5374 \glskeylisttok{##1}%
5375 \glslabeltok{##2}%
5376 \glsshorttok{##3}%
5377 \glslongtok{##4}%
5378 \newacronymhook
5379 \DescriptionNewAcronymDef
5380 }%

```

Set display.

```

5381 \@for\@gls@type:=\@glsacronymlists\do{%
5382 \SetDescriptionAcronymDisplayStyle{\@gls@type}%
5383 }%

```

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.

```

5384 \ifglsacrsmallcaps
5385 \renewcommand{\acronymfont}[1]{\textsc{##1}}
5386 \renewcommand*{\acrpluralsuffix}{%
5387 \glstextup{\glspluralsuffix}}%
5388 \else
5389 \ifglsacrsmaller
5390 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
5391 \fi
5392 \fi
5393 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

5394 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
5395   \def\glsentryfmt[#1]{%
      Move the inserted text outside of \acronymfont
5396     \let\gls@org@insert\glsinsert
5397     \let\glsinsert\@empty
5398     \ifglsused{\glslabel}%
5399     {%
5400       \acronymfont{\glsentryfmt}\gls@org@insert
5401     }%
5402     {%
5403       \firstacronymfont{\glsentryfmt}\gls@org@insert
5404       \ifglschaslong{\glslabel}%
5405       {%
5406         \expandafter\protect\expandafter\acrfootnote\expandafter
5407         {\@gls@link@opts}{\@gls@link@label}%
5408         {%
5409           \glsifplural
5410             {\glsentrylongpl{\glslabel}}%
5411             {\glsentrylong{\glslabel}}%
5412           }%
5413         }%
5414       }%
5415     }%
5416   }%
5417 }
```

FootnoteNewAcronymDef

```

5418 \newcommand*{\FootnoteNewAcronymDef}{%
5419   \edef\@do@newglossaryentry{%
5420     \noexpand\newglossaryentry{\the\glslabeltok}%
5421     {%
5422       type=\acronymtype,%
5423       name={\noexpand\acronymfont{\the\glsshorttok}},%
5424       sort={\the\glsshorttok},%
5425       text={\the\glsshorttok},%
5426       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5427       first={\the\glsshorttok},%
5428       firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5429       short={\the\glsshorttok},%
5430       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5431       long={\the\glslongtok},%
5432       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5433       description={\the\glslongtok},%
5434       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5435       \the\glskeylisttok
5436     }%
5437   }
```



```

5437 }%
5438 \let\@org@gl@s@assign@plural\gl@s@assign@plural
5439 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
5440 \let\@org@gl@s@assign@descplural\gl@s@assign@descplural
5441 \def\gl@s@assign@firstpl##1##2{%
5442   \@@gl@s@expand@field{##1}{firstpl}{##2}%
5443 }%
5444 \def\gl@s@assign@plural##1##2{%
5445   \@@gl@s@expand@field{##1}{plural}{##2}%
5446 }%
5447 \def\gl@s@assign@descplural##1##2{%
5448   \@@gl@s@expand@field{##1}{descplural}{##2}%
5449 }%
5450 \do@newglossaryentry
5451 \let\gl@s@assign@plural\@org@gl@s@assign@plural
5452 \let\gl@s@assign@firstpl\@org@gl@s@assign@firstpl
5453 \let\gl@s@assign@descplural\@org@gl@s@assign@descplural
5454 }

```

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.

```

5455 \newcommand*\SetFootnoteAcronymStyle{%
5456   \renewcommand{\newacronym}[4][{}]{%
5457     \ifx\@gl@s@acronymlists\@empty
5458       \def\@glo@type{\acronymtype}%
5459       \setkeys{glossentry}{##1}%
5460       \DeclareAcronymList{\@glo@type}%
5461       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
5462     \fi
5463     \gl@keylisttok{##1}%
5464     \gl@labeltok{##2}%
5465     \gl@shorttok{##3}%
5466     \gl@longtok{##4}%
5467     \newacronymhook
5468     \FootnoteNewAcronymDef
5469   }%

```

Set display

```

5470 \@for\@gl@s@type:=\@gl@s@acronymlists\do{%
5471   \SetFootnoteAcronymDisplayStyle{\@gl@s@type}%
5472 }%

```

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.

```

5473 \ifgl@acrsmallcaps
5474   \renewcommand*\acronymfont[1]{\textsc{##1}}%
5475   \renewcommand*\acrpluralsuffix{%
5476     \gl@textup{\gl@pluralsuffix}}%

```

```

5477 \else
5478     \ifglsmaller
5479         \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
5480     \fi
5481 \fi

    Check for option clash
5482 \ifglsmaller
5483     \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
5484     can’t both be set}{}%
5485 \fi
5486 }%

```

glsdoparenifnotempty Do a space followed by the argument if the argument doesn’t expand to empty or `\relax`. If argument isn’t empty (or `\relax`), apply the macro to it given in the second argument.

```

5487 \DeclareRobustCommand*{\glsdoparenifnotempty}[2]{%
5488     \protected@edef\gls@tmp{#1}%
5489     \ifdefempty\gls@tmp
5490     {}%
5491     {%
5492         \ifx\gls@tmp\@gls@default@value
5493         \else
5494             \space (#2{#1})%
5495         \fi
5496     }%
5497 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

5498 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
5499     \defglentryfmt[#1]{%

        Move the inserted text outside of \acronymfont
5500         \let\gls@org@insert\glsinsert
5501         \let\glsinsert\@empty
5502         \ifglused{\glslabel}%
5503         {%
5504             \acronymfont{\glsgenentryfmt}\gls@org@insert
5505         }%
5506         {%
5507             \glsgenentryfmt
5508             \ifglshassymbol{\glslabel}%
5509             {%
5510                 \glsifplural
5511                 {%
5512                     \def\@glo@symbol{\glsgenentrysymbolplural{\glslabel}}%
5513                 }%
5514                 {%
5515                     \def\@glo@symbol{\glsgenentrysymbol{\glslabel}}%

```

```

5516     }%
5517     \space
5518     (\glscapscase
5519     {\firstacronymfont{\@glo@symbol}}}%
5520     {\firstacronymfont{\@glo@symbol}}}%
5521     {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
5522     }%
5523     {}%
5524     }%
5525     }%
5526 }

```

\SmallNewAcronymDef

```

5527 \newcommand*{\SmallNewAcronymDef}{%
5528   \edef\@do@newglossaryentry{%
5529     \noexpand\newglossaryentry{\the\glslabeltok}%
5530     {%
5531       type=\acronymtype,%
5532       name={\noexpand\acronymfont{\the\glsshorttok}},%
5533       sort={\the\glsshorttok},%
5534       text={\the\glsshorttok},%

```

Default to the short plural.

```

5535     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5536     first={\the\glslongtok},%

```

Default to the long plural.

```

5537     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5538     short={\the\glsshorttok},%
5539     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5540     long={\the\glslongtok},%
5541     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5542     description={\noexpand\@glo@first},%
5543     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5544     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

5545     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5546     \the\glskeylisttok
5547   }%
5548 }%
5549 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5550 \let\@org@gls@assign@plural\gls@assign@plural
5551 \let\@org@gls@assign@descplural\gls@assign@descplural
5552 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5553 \def\gls@assign@firstpl##1##2{%
5554   \@@gls@expand@field{##1}{firstpl}{##2}%
5555 }%
5556 \def\gls@assign@plural##1##2{%
5557   \@@gls@expand@field{##1}{plural}{##2}%

```

```

5558 }%
5559 \def\gls@assign@descplural##1##2{%
5560   \@@gls@expand@field{##1}{descplural}{##2}%
5561 }%
5562 \def\gls@assign@symbolplural##1##2{%
5563   \@@gls@expand@field{##1}{symbolplural}{##2}%
5564 }%
5565 \do@newglossaryentry
5566 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5567 \let\gls@assign@plural\@org@gls@assign@plural
5568 \let\gls@assign@descplural\@org@gls@assign@descplural
5569 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5570 }

```

`\setSmallAcronymStyle` Neither footnote nor description required, but smallcaps or smaller specified.
 Use the symbol key to store the short form and first to store the long form.

```

5571 \newcommand*\setSmallAcronymStyle{%
5572   \renewcommand{\newacronym}[4][]{%
5573     \ifx\@glsacronymlists\@empty
5574       \def\@glo@type{\acronymtype}%
5575       \setkeys{glossentry}{##1}%
5576       \DeclareAcronymList{\@glo@type}%
5577       \SetSmallAcronymDisplayStyle{\@glo@type}%
5578     \fi
5579     \glskeylisttok{##1}%
5580     \glslabeltok{##2}%
5581     \glsshorttok{##3}%
5582     \gslongtok{##4}%
5583     \newacronymhook
5584     \SmallNewAcronymDef
5585   }%

```

Change the display since first only contains long form.

```

5586 \@for\@gls@type:=\@glsacronymlists\do{%
5587   \SetSmallAcronymDisplayStyle{\@gls@type}%
5588 }%

```

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.

```

5589 \ifglsacrsmallcaps
5590   \renewcommand*\acronymfont{[1]{\textsc{##1}}}
5591   \renewcommand*\acrpluralsuffix{%
5592     \glstextup{\glspluralsuffix}}%
5593 \else
5594   \renewcommand*\acronymfont{[1]{\textsmaller{##1}}}
5595 \fi

```

check for option clash

```

5596 \ifglsacrdua

```

```

5597 \ifglsmallcaps
5598 \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
5599 can’t both be set}{}%
5600 \else
5601 \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
5602 can’t both be set}{}%
5603 \fi
5604 \fi
5605 }%

```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```

5606 \newcommand*\SetDUADisplayStyle[1]{%
5607 \defglentryfmt[#1]{\glsgenentryfmt}%
5608 }

```

\DUANewAcronymDef

```

5609 \newcommand*\DUANewAcronymDef{%
5610 \edef\@do@newglossaryentry{%
5611 \noexpand\newglossaryentry{\the\glslabeltok}%
5612 {%
5613 type=\acronymtype,%
5614 name={\the\glsshorttok},%
5615 text={\the\glslongtok},%
5616 first={\the\glslongtok},%
5617 plural={\noexpand\expandonce\noexpand\@glo@longpl},%
5618 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5619 short={\the\glsshorttok},%
5620 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5621 long={\the\glslongtok},%
5622 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5623 description={\the\glslongtok},%
5624 descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5625 symbol={\the\glsshorttok},%
5626 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5627 \the\glskeylisttok
5628 }%
5629 }%
5630 \let\@org@glsgl@assign@firstpl\glsgl@assign@firstpl
5631 \let\@org@glsgl@assign@plural\glsgl@assign@plural
5632 \let\@org@glsgl@assign@symbolplural\glsgl@assign@symbolplural
5633 \let\@org@glsgl@assign@descplural\glsgl@assign@descplural
5634 \def\glsgl@assign@firstpl##1##2{%
5635 \@@glsgl@expand@field{##1}{firstpl}{##2}%
5636 }%
5637 \def\glsgl@assign@plural##1##2{%
5638 \@@glsgl@expand@field{##1}{plural}{##2}%
5639 }%
5640 \def\glsgl@assign@symbolplural##1##2{%
5641 \@@glsgl@expand@field{##1}{symbolplural}{##2}%

```

```

5642 }%
5643 \def\gls@assign@descplural##1##2{%
5644   \@@gls@expand@field{##1}{descplural}{##2}%
5645 }%
5646 \do@newglossaryentry
5647 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5648 \let\gls@assign@plural\@org@gls@assign@plural
5649 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5650 \let\gls@assign@descplural\@org@gls@assign@descplural
5651 }

```

`\SetDUASStyle` Always expand acronyms.

```

5652 \newcommand*{\SetDUASStyle}{%
5653   \renewcommand{\newacronym}[4][]{%
5654     \ifx\@glsacronymlists\@empty
5655       \def\@glo@type{\acronymtype}%
5656       \setkeys{glossentry}{##1}%
5657       \DeclareAcronymList{\@glo@type}%
5658       \SetDUADisplayStyle{\@glo@type}%
5659     \fi
5660     \glskeylisttok{##1}%
5661     \glslabeltok{##2}%
5662     \glsshorttok{##3}%
5663     \glslongtok{##4}%
5664     \newacronymhook
5665     \DUANewAcronymDef
5666   }%

```

Set the display

```

5667   \@for\@gls@type:=\@glsacronymlists\do{%
5668     \SetDUADisplayStyle{\@gls@type}%
5669   }%
5670 }

```

`\SetAcronymStyle`

```

5671 \newcommand*{\SetAcronymStyle}{%
5672   \SetDefaultAcronymStyle
5673   \ifglsacrdescription
5674     \ifglsacrfootnote
5675       \SetDescriptionFootnoteAcronymStyle
5676     \else
5677       \ifglsacrdua
5678         \SetDescriptionDUAAcronymStyle
5679       \else
5680         \SetDescriptionAcronymStyle
5681       \fi
5682     \fi
5683   \else
5684     \ifglsacrfootnote
5685       \SetFootnoteAcronymStyle

```

```

5686 \else
5687 \ifthenelse{\boolean{glsacrsmallcaps}}\OR
5688 \boolean{glsacrsmaller}}%
5689 {%
5690 \SetSmallAcronymStyle
5691 }%
5692 {%
5693 \ifglsacrdua
5694 \SetDUASstyle
5695 \fi
5696 }%
5697 \fi
5698 \fi
5699 }

```

Set the acronym style according to the package options

```
5700 \SetAcronymStyle
```

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

`\SetCustomDisplayStyle` Sets the acronym display style.

```

5701 \newcommand*\SetCustomDisplayStyle[1]{%
5702 \defglsentryfmt[#1]{\glsentryfmt}%
5703 }

```

`\CustomAcronymFields`

```

5704 \newcommand*\CustomAcronymFields{%
5705 name={\the\glsshorttok},%
5706 description={\the\glslongtok},%
5707 first={\noexpand\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
5708 firstplural={\noexpand\acrfullformat
5709 {\noexpand\glsentrylongpl{\the\glslabeltok}}%
5710 {\noexpand\glsentryshortpl{\the\glslabeltok}}}%
5711 text={\the\glsshorttok},%
5712 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
5713 }

```

`\CustomNewAcronymDef`

```

5714 \newcommand*\CustomNewAcronymDef{%
5715 \protected@edef\@do@newglossaryentry{%
5716 \noexpand\newglossaryentry{\the\glslabeltok}%
5717 {%
5718 type=\acronymtype,%
5719 short={\the\glsshorttok},%
5720 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%

```

```

5721     long={\the\glslongtok},%
5722     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5723     user1={\the\glsshorttok},%
5724     user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
5725     user3={\the\glslongtok},%
5726     user4={\the\glslongtok\noexpand\acrpluralsuffix},%
5727     \CustomAcronymFields,%
5728     \the\glskeylisttok
5729 }%
5730 }%
5731 \@do@newglossaryentry
5732 }

```

\SetCustomStyle

```

5733 \newcommand*{\SetCustomStyle}{%
5734   \renewcommand{\newacronym}[4][[]]{%
5735     \ifx\@glsacronymlists\@empty
5736       \def\@glo@type{\acronymtype}%
5737       \setkeys{glossentry}{##1}%
5738       \DeclareAcronymList{\@glo@type}%
5739       \SetCustomDisplayStyle{\@glo@type}%
5740     \fi
5741     \glskeylisttok{##1}%
5742     \glslabeltok{##2}%
5743     \glsshorttok{##3}%
5744     \glslongtok{##4}%
5745     \newacronymhook
5746     \CustomNewAcronymDef
5747   }%

```

Set the display

```

5748   \@for\@gls@type:=\@glsacronymlists\do{%
5749     \SetCustomDisplayStyle{\@gls@type}%
5750   }%
5751 }

```

fineAcronymSynonyms

```

5752 \newcommand*{\DefineAcronymSynonyms}{%

```

Short form

\acs

```

5753   \let\acs\acrshort

```

First letter uppercase short form

\Acs

```

5754   \let\Acs\Acrshort

```

Plural short form

`\acsp`
 5755 `\let\acsp\acrshortpl`
 First letter uppercase plural short form

`\Acsp`
 5756 `\let\Acsp\Acrshortpl`
 Long form

`\acl`
 5757 `\let\acl\acrlong`
 Plural long form

`\aclp`
 5758 `\let\aclp\acrlongpl`
 First letter upper case long form

`\Acl`
 5759 `\let\Acl\Acrlong`
 First letter upper case plural long form

`\Aclp`
 5760 `\let\Aclp\Acrlongpl`
 Full form

`\acf`
 5761 `\let\acf\acrfull`
 Plural full form

`\acfp`
 5762 `\let\acfp\acrfullpl`
 First letter upper case full form

`\Acf`
 5763 `\let\Acf\Acrfull`
 First letter upper case plural full form

`\Acfp`
 5764 `\let\Acfp\Acrfullpl`
 Standard form

`\ac`
 5765 `\let\ac\gls`

First upper case standard form

`\Ac`

```
5766 \let\Ac\Gls
```

Standard plural form

`\acp`

```
5767 \let\acp\glspl
```

Standard first letter upper case plural form

`\Acp`

```
5768 \let\Acp\Glspl
```

```
5769 }
```

Define synonyms if required

```
5770 \ifglsacrshortcuts
```

```
5771 \DefineAcronymSynonyms
```

```
5772 \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.

```
5773 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
5774 \@gls@loadlist
```

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

```
5775 \@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.

```
5776 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the `notree` package option is used.

```
5777 \@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.

```
5778 \ifx\@glossary@default@style\relax
```

```
5779 \else
```

```
5780 \setglossarystyle{\@glossary@default@style}
```

```
5781 \fi
```

1.19 Debugging Commands

`\showgloparent` `\showgloparent{<label>}`

```
5782 \newcommand*{\showgloparent}[1]{%
5783   \expandafter\show\csname glo@#1@parent\endcsname
5784 }
```

`\showglolevel` `\showglolevel{<label>}`

```
5785 \newcommand*{\showglolevel}[1]{%
5786   \expandafter\show\csname glo@#1@level\endcsname
5787 }
```

`\showglotext` `\showglotext{<label>}`

```
5788 \newcommand*{\showglotext}[1]{%
5789   \expandafter\show\csname glo@#1@text\endcsname
5790 }
```

`\showgloplural` `\showgloplural{<label>}`

```
5791 \newcommand*{\showgloplural}[1]{%
5792   \expandafter\show\csname glo@#1@plural\endcsname
5793 }
```

`\showglofirst` `\showglofirst{<label>}`

```
5794 \newcommand*{\showglofirst}[1]{%
5795   \expandafter\show\csname glo@#1@first\endcsname
5796 }
```

`\showglofirstpl` `\showglofirstpl{<label>}`

```
5797 \newcommand*{\showglofirstpl}[1]{%
5798   \expandafter\show\csname glo@#1@firstpl\endcsname
5799 }
```

`\showglotype` `\showglotype{<label>}`

```

5800 \newcommand*{\showglotype}[1]{%
5801   \expandafter\show\csname glo@#1@type\endcsname
5802 }

```

\showglocounter \showglocounter{<label>}

```

5803 \newcommand*{\showglocounter}[1]{%
5804   \expandafter\show\csname glo@#1@counter\endcsname
5805 }

```

\showglouser \showglouser{<label>}

```

5806 \newcommand*{\showglouser}[1]{%
5807   \expandafter\show\csname glo@#1@useri\endcsname
5808 }

```

\showglouserii \showglouserii{<label>}

```

5809 \newcommand*{\showglouserii}[1]{%
5810   \expandafter\show\csname glo@#1@userii\endcsname
5811 }

```

\showglouseriii \showglouseriii{<label>}

```

5812 \newcommand*{\showglouseriii}[1]{%
5813   \expandafter\show\csname glo@#1@useriii\endcsname
5814 }

```

\showglouseriv \showglouseriv{<label>}

```

5815 \newcommand*{\showglouseriv}[1]{%
5816   \expandafter\show\csname glo@#1@useriv\endcsname
5817 }

```

\showglouserv \showglouserv{<label>}

```

5818 \newcommand*{\showglouserv}[1]{%
5819   \expandafter\show\csname glo@#1@userv\endcsname
5820 }

```

\showglouservi \showglouservi{<label>}

```
5821 \newcommand*{\showglouservi}[1]{%
5822   \expandafter\show\csname glo@#1@uservi\endcsname
5823 }
```

\showgloname \showgloname{<label>}

```
5824 \newcommand*{\showgloname}[1]{%
5825   \expandafter\show\csname glo@#1@name\endcsname
5826 }
```

\showglodesc \showglodesc{<label>}

```
5827 \newcommand*{\showglodesc}[1]{%
5828   \expandafter\show\csname glo@#1@desc\endcsname
5829 }
```

\showglodescplural \showglodescplural{<label>}

```
5830 \newcommand*{\showglodescplural}[1]{%
5831   \expandafter\show\csname glo@#1@descplural\endcsname
5832 }
```

\showglosort \showglosort{<label>}

```
5833 \newcommand*{\showglosort}[1]{%
5834   \expandafter\show\csname glo@#1@sort\endcsname
5835 }
```

\showglosymbol \showglosymbol{<label>}

```
5836 \newcommand*{\showglosymbol}[1]{%
5837   \expandafter\show\csname glo@#1@symbol\endcsname
5838 }
```

\showglosymbolplural \showglosymbolplural{<label>}

```
5839 \newcommand*{\showglosymbolplural}[1]{%
5840   \expandafter\show\csname glo@#1@symbolplural\endcsname
5841 }
```

`\showgloshort` `\showgloshort{<label>}`

```
5842 \newcommand*{\showgloshort}[1]{%
5843   \expandafter\show\csname glo@#1@short\endcsname
5844 }
```

`\showglolong` `\showglolong{<label>}`

```
5845 \newcommand*{\showglolong}[1]{%
5846   \expandafter\show\csname glo@#1@long\endcsname
5847 }
```

`\showgloindex` `\showgloindex{<label>}`

```
5848 \newcommand*{\showgloindex}[1]{%
5849   \expandafter\show\csname glo@#1@index\endcsname
5850 }
```

`\showgloflag` `\showgloflag{<label>}`

```
5851 \newcommand*{\showgloflag}[1]{%
5852   \expandafter\show\csname ifglo@#1@flag\endcsname
5853 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
5854 \newcommand*{\showacronymlists}{%
5855   \show\@glsacronymlists
5856 }
```

`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
5857 \newcommand*{\showglossaries}{%
5858   \show\@glo@types
5859 }
```

`\showglossaryin` `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
5860 \newcommand*{\showglossaryin}[1]{%
5861   \expandafter\show\csname @glotype@#1@in\endcsname
5862 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
5863 \newcommand*{\showglossaryout}[1]{%
5864   \expandafter\show\csname @glotype@#1@out\endcsname
5865 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
5866 \newcommand*{\showglossarytitle}[1]{%
5867   \expandafter\show\csname @glotype@#1@title\endcsname
5868 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
5869 \newcommand*{\showglossarycounter}[1]{%
5870   \expandafter\show\csname @glotype@#1@counter\endcsname
5871 }
```

`\showglossaryentries` `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
5872 \newcommand*{\showglossaryentries}[1]{%
5873   \expandafter\show\csname glolist@#1\endcsname
5874 }
```

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.

```
5875 \csname ifglscpatible-2.07\endcsname
5876 \RequirePackage{glossaries-compatible-207}
5877 \fi
```

2 Prefix Support (glossaries-prefix Code)

This package provides a means of adding prefixes to your glossary entries. For example, you may want to use “a `\gls{<label>}`” on first use but use “an `\gls{<label>}`” on subsequent use.

```
5878 \NeedsTeXFormat{LaTeX2e}
5879 \ProvidesPackage{glossaries-prefix}[2013/11/14 v4.0 (NLCT)]
```

Pass all options to glossaries:

```
5880 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
5881 \ProcessOptions
```

Load glossaries:

```
5882 \RequirePackage{glossaries}
```

Add the new keys:

```
5883 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
5884 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
5885 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
5886 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%
```

Add them to `\@gls@keymap`:

```
5887 \appto\@gls@keymap{,%
5888   {prefixfirst}{prefixfirst},%
5889   {prefixfirstplural}{prefixfirstplural},%
5890   {prefix}{prefix},%
5891   {prefixplural}{prefixplural}}%
5892 }
```

Set the default values:

```
5893 \appto\@newglossaryentryprehook{%
5894   \def\@glo@entryprefix{}}%
5895   \def\@glo@entryprefixplural{}}%
5896   \let\@glo@entryprefixfirst\@gls@default@value
5897   \let\@glo@entryprefixfirstplural\@gls@default@value
5898 }
```


Set the assignment code:

```
5899 \appto\@newglossaryentryposthook{%
5900   \gls@assign@field{\@glo@label}{prefix}{\@glo@entryprefix}%
5901   \gls@assign@field{\@glo@label}{prefixplural}{\@glo@entryprefixplural}%
```

If prefixfirst has not been supplied, make it the same as prefix.

```
5902   \expandafter\gls@assign@field\expandafter
5903     {\csname glo@\@glo@label @prefix\endcsname}{\@glo@label}{prefixfirst}%
5904     {\@glo@entryprefixfirst}%
```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```
5905   \expandafter\gls@assign@field\expandafter
5906     {\csname glo@\@glo@label @prefixplural\endcsname}{\@glo@label}%
5907     {prefixfirstplural}{\@glo@entryprefixfirstplural}%
5908 }
```

Define commands to access these fields:

glsentryprefixfirst

```
5909 \newcommand*\glsentryprefixfirst[1]{\csuse{glo@#1@prefixfirst}}
```

ryprefixfirstplural

```
5910 \newcommand*\glsentryprefixfirstplural[1]{\csuse{glo@#1@prefixfirstplural}}
```

\glsentryprefix

```
5911 \newcommand*\glsentryprefix[1]{\csuse{glo@#1@prefix}}
```

lsentryprefixplural

```
5912 \newcommand*\glsentryprefixplural[1]{\csuse{glo@#1@prefixplural}}
```

Now for the initial upper case variants:

Glsentryprefixfirst

```
5913 \newrobustcmd*\Glsentryprefixfirst[1]{%
5914   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%
5915   \xmakefirstuc\@glo@text
5916 }
```

ryprefixfirstplural

```
5917 \newrobustcmd*\Glsentryprefixfirstplural[1]{%
5918   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
5919   \xmakefirstuc\@glo@text
5920 }
```

\Glsentryprefix

```
5921 \newrobustcmd*\Glsentryprefix[1]{%
5922   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
5923   \xmakefirstuc\@glo@text
5924 }
```

lsentryprefixplural

```
5925 \newrobustcmd*{\Glsentryprefixplural}[1]{%
5926   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
5927   \xmakefirstuc\@glo@text
5928 }
```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```
5929 \newcommand*{\ifglshasprefix}[3]{%
5930   \ifcempty{glo@#1@prefix}%
5931   {#3}%
5932   {#2}%
5933 }
```

ifglshasprefixplural

```
5934 \newcommand*{\ifglshasprefixplural}[3]{%
5935   \ifcempty{glo@#1@prefixplural}%
5936   {#3}%
5937   {#2}%
5938 }
```

ifglshasprefixfirst

```
5939 \newcommand*{\ifglshasprefixfirst}[3]{%
5940   \ifcempty{glo@#1@prefixfirst}%
5941   {#3}%
5942   {#2}%
5943 }
```

asprefixfirstplural

```
5944 \newcommand*{\ifglshasprefixfirstplural}[3]{%
5945   \ifcempty{glo@#1@prefixfirstplural}%
5946   {#3}%
5947   {#2}%
5948 }
```

Define commands that insert the prefix before commands like \gls:

\pgls

```
5949 \newrobustcmd{\pgls}{\@ifstar\@spgls\@pgls}
```

\@spgls Starred version.

```
5950 \newcommand*{\@spgls}[2] [] {\@pgls@{hyper=false,#1}{#2}}
```

\@pgls Unstarred version.

```
5951 \newcommand*{\@pgls}[2] [] {%
5952   \new@ifnextchar[%
5953   {\@pgls@{#1}{#2}}%
5954   {\@pgls@{#1}{#2} []}%
5955 }
```

\@pgls@ Read in the final optional argument:

```
5956 \def\@pgls@#1#2[#3]{%
5957   \glsdoifexists{#2}%
5958   {%
5959     \ifglsused{#2}%
5960     {%
5961       \glsentryprefix{#2}%
5962     }%
5963     {%
5964       \glsentryprefixfirst{#2}%
5965     }%
5966     \@pgls@{#1}{#2}[#3]%
5967   }%
5968 }
```

Similarly for the plural version:

\pglsp1

```
5969 \newrobustcmd{\pglsp1}{\@ifstar\@spglsp1\@pglsp1}
```

\@spglsp1 Starred version.

```
5970 \newcommand*{\@spglsp1}[2][\@pglsp1@{hyper=false,#1}{#2}]
```

\@pglsp1 Unstarred version.

```
5971 \newcommand*{\@pglsp1}[2][{%
5972   \new@ifnextchar[%
5973   {\@pglsp1@{#1}{#2}}%
5974   {\@pglsp1@{#1}{#2}[]}%
5975 }
```

\@pglsp1@ Read in the final optional argument:

```
5976 \def\@pglsp1@#1#2[#3]{%
5977   \glsdoifexists{#2}%
5978   {%
5979     \ifglsused{#2}%
5980     {%
5981       \glsentryprefixplural{#2}%
5982     }%
5983     {%
5984       \glsentryprefixfirstplural{#2}%
5985     }%
5986     \@pglsp1@{#1}{#2}[#3]%
5987   }%
5988 }
```

Now for the first letter upper case versions:

\Pgls

```
5989 \newrobustcmd{\Pgls}{\@ifstar\@sPgls\@Pgls}
```

\@sPglS Starred version.

```
5990 \newcommand*{\@sPglS}[2][\@PglS@{hyper=false,#1}{#2}]
```

\@PglS Unstarred version.

```
5991 \newcommand*{\@PglS}[2][\%
5992 \new@ifnextchar[\%
5993 {\@PglS@{#1}{#2}}\%
5994 {\@PglS@{#1}{#2}[]}\%
5995 }
```

\@PglS@ Read in the final optional argument:

```
5996 \def\@PglS@#1#2[#3]{\%
5997 \glsdoifexists{#2}\%
5998 {\%
5999 \ifglSused{#2}\%
6000 {\%
6001 \ifglshasprefix{#2}\%
6002 {\%
6003 \Glsentryprefix{#2}\%
6004 \@glS@{#1}{#2}[#3]\%
6005 }%
6006 {\@Gls@{#1}{#2}[#3]}\%
6007 }%
6008 {\%
6009 \ifglshasprefixfirst{#2}\%
6010 {\%
6011 \Glsentryprefixfirst{#2}\%
6012 \@glS@{#1}{#2}[#3]\%
6013 }%
6014 {\@Gls@{#1}{#2}[#3]}\%
6015 }%
6016 }%
6017 }
```

Similarly for the plural version:

\PglSpl

```
6018 \newrobustcmd{\PglSpl}{\@ifstar\@sPglSpl\@PglSpl}
```

\@sPglSpl Starred version.

```
6019 \newcommand*{\@sPglSpl}[2][\@PglSpl@{hyper=false,#1}{#2}]
```

\@PglSpl Unstarred version.

```
6020 \newcommand*{\@PglSpl}[2][\%
6021 \new@ifnextchar[\%
6022 {\@PglSpl@{#1}{#2}}\%
6023 {\@PglSpl@{#1}{#2}[]}\%
6024 }
```

\@Pglsp1@ Read in the final optional argument:

```
6025 \def\@Pglsp1@#1#2[#3]{%
6026   \glsdoifexists{#2}%
6027   {%
6028     \ifglused{#2}%
6029     {%
6030       \ifglshasprefixplural{#2}%
6031       {%
6032         \Glsentryprefixplural{#2}%
6033         \@glsp1@{#1}{#2}[#3]%
6034       }%
6035       {\@Glspl@{#1}{#2}[#3]}%
6036     }%
6037     {%
6038       \ifglshasprefixfirstplural{#2}%
6039       {%
6040         \Glsentryprefixfirstplural{#2}%
6041         \@glsp1@{#1}{#2}[#3]%
6042       }%
6043       {\@Glspl@{#1}{#2}[#3]}%
6044     }%
6045   }%
6046 }
```

Finally the all upper case versions:

\PGLS

```
6047 \newrobustcmd{\PGLS}{\@ifstar\@sPGLS\@PGLS}
```

\@sPGLS Starred version.

```
6048 \newcommand*{\@sPGLS}[2][\@PGLS@{hyper=false,#1}{#2}]
```

\@PGLS Unstarred version.

```
6049 \newcommand*{\@PGLS}[2][\@PGLS@{hyper=false,#1}{#2}]
6050 \new@ifnextchar[%
6051 {\@PGLS@{#1}{#2}}%
6052 {\@PGLS@{#1}{#2}[]}%
6053 }
```

\@PGLS@ Read in the final optional argument:

```
6054 \def\@PGLS@#1#2[#3]{%
6055   \glsdoifexists{#2}%
6056   {%
6057     \ifglused{#2}%
6058     {%
6059       \mfirstucMakeUppercase{\glsentryprefix{#2}}%
6060     }%
6061     {%
6062       \mfirstucMakeUppercase{\glsentryprefixfirst{#2}}%
6063     }%
6064   }%
6065 }
```

```

6063 }%
6064 \@GLS@{#1}{#2}[#3]%
6065 }%
6066 }

```

Plural version:

\PGLSp1

```
6067 \newrobustcmd{\PGLSp1}{\@ifstar\@sPGLSp1\@PGLSp1}
```

\@sPGLSp1 Starred version.

```
6068 \newcommand*{\@sPGLSp1}[2][\@PGLSp1@{hyper=false,#1}{#2}]
```

\@PGLSp1 Unstarred version.

```

6069 \newcommand*{\@PGLSp1}[2][\@PGLSp1@{hyper=false,#1}{#2}]
6070 \new@ifnextchar[%
6071 {\@PGLSp1@{#1}{#2}}%
6072 {\@PGLSp1@{#1}{#2}}%
6073 }

```

\@PGLSp1@ Read in the final optional argument:

```

6074 \def\@PGLSp1@#1#2[#3]{%
6075 \glsdoifexists{#2}%
6076 {%
6077 \ifglsused{#2}%
6078 {%
6079 \mfirstucMakeUppercase{\glsentryprefixplural{#2}}%
6080 }%
6081 {%
6082 \mfirstucMakeUppercase{\glsentryprefixfirstplural{#2}}%
6083 }%
6084 \@GLSp1@{#1}{#2}[#3]%
6085 }%
6086 }

```

3 Mfirstuc Documented Code

```

6087 \NeedsTeXFormat{LaTeX2e}
6088 \ProvidesPackage{mfirstuc}[2013/11/04 v1.08 (NLCT)]

```

Requires etoolbox:

```
6089 \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`.

```

6090 \newif\if@glscs
6091 \newtoks\@glsmfirst
6092 \newtoks\@glsmrest
6093 \newrobustcmd*{\makefirstuc}[1]{%
6094   \def\gls@argi{#1}%
6095   \ifx\gls@argi\@empty

  If the argument is empty, do nothing.

6096   \else

6097     \def\@gls@tmp{\ #1}%
6098     \@onelevel@sanitize\@gls@tmp
6099     \expandafter\@gls@checkcs\@gls@tmp\relax\relax
6100     \if@glscs
6101       \@gls@getbody #1{}\@nil
6102       \ifx\@gls@rest\@empty
6103         \glsmakefirstuc{#1}%
6104       \else
6105         \expandafter\@gls@split\@gls@rest\@nil
6106         \ifx\@gls@first\@empty
6107           \glsmakefirstuc{#1}%
6108         \else
6109           \expandafter\@glsmfirst\expandafter{\@gls@first}%
6110           \expandafter\@glsmrest\expandafter{\@gls@rest}%
6111           \edef\@gls@domfirstuc{\noexpand\@gls@body
6112             {\noexpand\glsmakefirstuc\the\@glsmfirst}%
6113             \the\@glsmrest}%
6114           \@gls@domfirstuc
6115         \fi
6116       \fi
6117     \else
6118       \glsmakefirstuc{#1}%
6119     \fi
6120 \fi
6121 }
```

Put first argument in `\@gls@first` and second argument in `\@gls@rest`:

```

6122 \def\@gls@split#1#2\@nil{%
6123   \def\@gls@first{#1}\def\@gls@rest{#2}%
6124 }

6125 \def\@gls@checkcs#1 #2#3\relax{%
6126   \def\@gls@argi{#1}\def\@gls@argii{#2}%
6127   \ifx\@gls@argi\@gls@argii
6128     \@glscstrue
6129   \else
6130     \@glscsfalse
```

```
6131 \fi
6132 }
```

`\@gls@makefirstuc` Make first thing upper case:

```
6133 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}
```

`\mfirstucMakeUppercase` Allow user to replace `\MakeUppercase` with another case changing command.

```
6134 \newcommand*\mfirstucMakeUppercase{\MakeUppercase}
```

`\glsmakefirstuc` Provide a user command to make it easier to customise.

```
6135 \newcommand*\glsmakefirstuc[1]{\@gls@makefirstuc{#1}}
```

Get the first grouped argument and stores in `\@gls@body`.

```
6136 \def\@gls@getbody#1#\def\@gls@body{#1}\@gls@gobbletonil}
```

Scoup up everything to `\@nil` and store in `\@gls@rest`:

```
6137 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}
```

`\xmakefirstuc` Expand argument once before applying `\makefirstuc` (added v1.01).

```
6138 \newcommand*\xmakefirstuc[1]{%
```

```
6139 \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).

```
6140 \newrobustcmd*\capitalisewords[1]{%
```

```
6141 \def\gls@add@space{ }
```

```
6142 \mfu@capitalisewords#1 \@nil\mfu@endcap
```

```
6143 }
```

```
6144 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
```

```
6145 \def\mfu@cap@first{#1}%
```

```
6146 \def\mfu@cap@second{#2}%
```

```
6147 \gls@add@space
```

```
6148 \makefirstuc{#1}%
```

```
6149 \def\gls@add@space{ }
```

```
6150 \ifx\mfu@cap@second\@nnil
```

```
6151 \let\next\mfu@cap\mfu@noop
```

```
6152 \else
```

```
6153 \let\next\mfu@cap\mfu@capitalisewords
```

```
6154 \fi
```

```
6155 \next\mfu@cap#2\mfu@endcap
```

```
6156 }
```

```
6157 \def\mfu@noop#1\mfu@endcap{ }
```

`\xcapitalisewords` Short-cut command:

```
6158 \newcommand*\xcapitalisewords[1]{%
```

```
6159 \expandafter\capitalisewords\expandafter{#1}%
```

```
6160 }
```


4 Glossary Styles

4.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```
6161 \ProvidesPackage{glossary-hypernav}[2013/11/14 v4.0 (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
```

```
6162 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
6163   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
6164   \@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
```

```
6165 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
6166   \protected@write\@auxout{}{\string\@gls@hypergroup{#1}{#2}}%
  Add the target.
6167   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
6168   \expandafter\let
6169     \expandafter\@gls@list\csname @gls@hypergroup@#1\endcsname
  Iterate through list and terminate loop if this group is found.
6170   \@for\@gls@elem:=\@gls@list\do{%
6171     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}%
  Check if list terminated prematurely.
6172   \if@endfor
6173   \else
```

This group was not included in the list, so issue a warning.

```

6174 \GlossariesWarningNoLine{Navigation panel
6175     for glossary type ‘#1’^^Jmissing group ‘#2’}%
6176 \gdef\gls@hypergroupprerun{%
6177     \GlossariesWarningNoLine{Navigation panel
6178     has changed. Rerun LaTeX}}%
6179 \fi
6180 }

```

`\gls@hypergroupprerun` Give a warning at the end if re-run required

```

6181 \let\gls@hypergroupprerun\relax
6182 \AtEndDocument{\gls@hypergroupprerun}

```

`\@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.

```

6183 \newcommand*{\@gls@hypergroup}[2]{%
6184 \@ifundefined{\@gls@hypergrouplist@#1}{%
6185     \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{#2}%
6186 }{%
6187     \expandafter\let\expandafter\@gls@tmp
6188     \csname @gls@hypergrouplist@#1\endcsname
6189     \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{%
6190         \@gls@tmp,#2}%
6191 }%
6192 }

```

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`

```

6193 \newcommand*{\glsnavigation}{%
6194 \def\@gls@between{}%
6195 \@ifundefined{\@gls@hypergrouplist@\@glo@type}{%
6196     \def\@gls@list{}%
6197 }{%
6198     \expandafter\let\expandafter\@gls@list
6199     \csname @gls@hypergrouplist@\@glo@type\endcsname
6200 }%
6201 \@for\@gls@tmp:=\@gls@list\do{%
6202     \@gls@between
6203     \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%

```

```

6204 \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
6205 \let\@gls@between\glshypernavsep%
6206 }%
6207 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```

6208 \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`

```

6209 \newcommand*{\glssymbolnav}{%
6210 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
6211 \glshypernavsep
6212 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
6213 \glshypernavsep
6214 }

```

4.2 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.

```

6215 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]

```

`inline` Define the inline style.

```

6216 \newglossarystyle{inline}{%

```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```

6217 \renewenvironment{theglossary}%
6218 {%
6219 \def\gls@inlinesep{}%
6220 \def\gls@inlinesubsep{}%
6221 \def\gls@inlinepostchild{}%
6222 }%
6223 {\glspostinline}%

```

No header:

```

6224 \renewcommand*{\glossaryheader}{}%

```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```

6225 \renewcommand*{\glsgroupheading}[1]{}%

```

Just display separator followed by name and description:

```

6226 \renewcommand{\glossentry}[2]{%
6227 \glsinlinedopostchild
6228 \gls@inlinesep

```

```

6229 \glsentryitem{##1}%
6230 \glsinlinenameformat{##1}{%
6231 \glossentryname{##1}%
6232 }%
6233 \ifglstdescsuppressed{##1}%
6234 {%
6235 \glsinlineemptydescformat
6236 {%
6237 \glossentrysymbol{##1}%
6238 }%
6239 {%
6240 ##2%
6241 }%
6242 }%
6243 {%
6244 \ifglshasdesc{##1}%
6245 {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
6246 {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
6247 }%
6248 \ifglshaschildren{##1}%
6249 {%
6250 \glsresetsubentrycounter
6251 \glsinlineparentchildseparator
6252 \def\gls@inlinesubsep{}%
6253 \def\gls@inlinepostchild{\glsinlinepostchild}%
6254 }%
6255 {}%
6256 \def\gls@inlinesep{\glsinlineseparator}%
6257 }%

```

Sub-entries display description:

```

6258 \renewcommand{\subglossentry}[3]{%
6259 \gls@inlinesubsep%
6260 \glsinlinesubnameformat{##2}{%
6261 \glossentryname{##2}}%
6262 \glssubentryitem{##2}%
6263 \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
6264 \def\gls@inlinesubsep{\glsinlinesubseparator}%
6265 }%

```

Nothing special between groups:

```

6266 \renewcommand*{\glsgroupskip}{}%
6267 }

```

`\glsinlinedopostchild`

```

6268 \newcommand*{\glsinlinedopostchild}{%
6269 \gls@inlinepostchild
6270 \def\gls@inlinepostchild{}%
6271 }

```

`\glsinlineseparator` Separator to use between entries.
6272 `\newcommand*\glsinlineseparator}{;\space}`

`\glsinlinesubseparator` Separator to use between sub-entries.
6273 `\newcommand*\glsinlinesubseparator}{,\space}`

`\glsinlineparentchildseparator` Separator to use between parent and children.
6274 `\newcommand*\glsinlineparentchildseparator}{:\space}`

`\glsinlinepostchild` Hook to use between child and next entry
6275 `\newcommand*\glsinlinepostchild}{}`

`\glspostinline` Terminator for inline glossary.
6276 `\newcommand*\glspostinline}{\glspostdescription\space}`

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):
6277 `\newcommand*\glsinlinenameformat}[2]{\glstarget{#1}{#2}}`

`\glsinlinedescformat` Formats the entry's description, symbol and location list:
6278 `\newcommand*\glsinlinedescformat}[3]{\space#1}`

`\glsinlineemptydescformat` Formats the entry's symbol and location list when the description is empty:
6279 `\newcommand*\glsinlineemptydescformat}[2]{}`

`\glsinlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):
6280 `\newcommand*\glsinlinesubnameformat}[2]{\glstarget{#1}{}}`

`\glsinlinesubdescformat` Formats the subentry's description, symbol and location list:
6281 `\newcommand*\glsinlinesubdescformat}[3]{#1}`

4.3 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.

6282 `\ProvidesPackage{glossary-list}[2013/11/14 v4.0 (NLCT)]`

`list` The list glossary style uses the description environment. The group separator `\glsgroupskip` is redefined as `\indexspace` which produces a gap between groups. The glossary heading and the group headings do nothing. Sub-entries immediately follow the main entry without the sub-entry name. This style does not use the entry's symbol. This is used as the default style for the glossaries package.

6283 `\newglossarystyle{list}{%`

Use description environment:

```
6284 \renewenvironment{theglossary}%  
6285   {\begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
6286 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6287 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
6288 \renewcommand*{\glossentry}[2]{%  
6289   \item[\glsentryitem{##1}]%  
6290       \glstarget{##1}{\glossentryname{##1}}]  
6291   \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries continue on the same line:

```
6292 \renewcommand*{\subglossentry}[3]{%  
6293   \glssubentryitem{##2}%  
6294   \glstarget{##2}{\strut}%  
6295   \glossentrydesc{##2}\glspostdescription\space ##3.}%  
6296 % \end{macrocode}  
6297 % Add vertical space between groups:  
6298 %\changes{3.03}{2012/09/21}{added check for glsnogroupskip}  
6299 % \begin{macrocode}  
6300 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%  
6301 }
```

listgroup The listgroup style is like the list style, but the glossary groups have headings.

```
6302 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
6303 \setglossarystyle{list}%
```

Each group has a heading:

```
6304 \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.

```
6305 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
6306 \setglossarystyle{list}%
```

Add navigation links at the start of the environment:

```
6307 \renewcommand*{\glossaryheader}{%  
6308   \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
6309 \renewcommand*{\glsgroupheading}[1]{%  
6310   \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.

```
6311 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
6312 \setglossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
6313 \renewcommand*{\glossentry}[2]{%
6314 \item[\glsentryitem{##1}%
6315 \glstarget{##1}{\glossentryname{##1}}}%
```

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.

```
6316 \mbox{}\par\nobreak\@afterheading
6317 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries start a new paragraph:

```
6318 \renewcommand{\subglossentry}[3]{%
6319 \par
6320 \glssubentryitem{##2}%
6321 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%
6322 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
6323 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
6324 \setglossarystyle{altlist}%
```

Each group has a heading:

```
6325 \renewcommand*{\glsgroupheading}[1]{\item[\glsgrouptitle{##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.

```
6326 \newglossarystyle{altlisthypergroup}{%
```

Base it on the altlist style:

```
6327 \setglossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
6328 \renewcommand*{\glossaryheader}{%
6329 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
6330 \renewcommand*{\glsgroupheading}[1]{%
6331 \item[\glsnavhypertarget{##1}{\glsgrouptitle{##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.

```
6332 \newglossarystyle{listdotted}{%
```

Base it on the list style:

```
6333 \setglossarystyle{list}{%
```

Each main (level 0) entry starts a new item:

```
6334 \renewcommand*{\glossentry}[2]{%
6335 \item[]\makebox[\glslistdottedwidth][l]{%
6336 \glsentryitem{##1}%
6337 \glstarget{##1}{\glossentryname{##1}}%
6338 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
6339 \renewcommand*{\subglossentry}[3]{%
6340 \item[]\makebox[\glslistdottedwidth][l]{%
6341 \glssubentryitem{##2}%
6342 \glstarget{##2}{\glossentryname{##2}}%
6343 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
6344 }
```

`\glslistdottedwidth`

```
6345 \newlength\glslistdottedwidth
6346 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
6347 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
6348 \setglossarystyle{listdotted}{%
```

Main (level 0) entries just display the name:

```
6349 \renewcommand*{\glossentry}[2]{%
6350 \item[\glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}}}%
6351 }
```

4.4 Glossary Styles using `longtable` (the `glossary-long` package)

The glossary styles defined in the package used the `longtable` environment in the glossary.

```
6352 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
6353 \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`.)

```
6354 \@ifundefined{glsdescwidth}{%
6355   \newlength{glsdescwidth
6356   \setlength{glsdescwidth}{0.6\hsize}
6357 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
6358 \@ifundefined{glspagelistwidth}{%
6359   \newlength{glspagelistwidth
6360   \setlength{glspagelistwidth}{0.1\hsize}
6361 }{}
```

`long` The long glossary style command which uses the `longtable` environment:

```
6362 \newglossarystyle{long}{%
```

Use `longtable` with two columns:

```
6363   \renewenvironment{theglossary}%
6364     {\begin{longtable}{lp{glsdescwidth}}}%
6365     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
6366   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
6367   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
6368   \renewcommand{\glossentry}[2]{%
6369     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6370     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
6371   }%
```

Sub entries displayed on the following row without the name:

```
6372   \renewcommand{\subglossentry}[3]{%
6373     &
6374     \glssubentryitem{##2}%
6375     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
6376     ##3\tabularnewline
6377   }%
```

Blank row between groups:

```
6378   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
6379   \tabularnewline\fi}%
6380 }
```

`longborder` The `longborder` style is like the above, but with horizontal and vertical lines:

```
6381 \newglossarystyle{longborder}{%
```

Base it on the `glostylelong` style:

```
6382 \setglossarystyle{long}%
```

Use `longtable` with two columns with vertical lines between each column:

```
6383 \renewenvironment{theglossary}{%
6384   \begin{longtable}{|l|p{\glstdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
6385 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6386 }
```

`longheader` The `longheader` style is like the `long` style but with a header:

```
6387 \newglossarystyle{longheader}{%
```

Base it on the `glostylelong` style:

```
6388 \setglossarystyle{long}%
```

Set the table's header:

```
6389 \renewcommand*{\glossaryheader}{%
6390   \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%
6391 }
```

`longheaderborder` The `longheaderborder` style is like the `long` style but with a header and border:

```
6392 \newglossarystyle{longheaderborder}{%
```

Base it on the `glostylelongborder` style:

```
6393 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
6394 \renewcommand*{\glossaryheader}{%
6395   \hline\bfseries \entryname & \bfseries
6396   \descriptionname\tabularnewline\hline
6397   \endhead
6398   \hline\endfoot}%
6399 }
```

`long3col` The `long3col` style is like `long` but with 3 columns

```
6400 \newglossarystyle{long3col}{%
```

Use a `longtable` with 3 columns:

```
6401 \renewenvironment{theglossary}%
6402   {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
6403   {\end{longtable}}%
```

No table header:

```
6404 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
6405 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
6406 \renewcommand{\glossentry}[2]{%
6407   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6408   \glossentrydesc{##1} & ##2\tabularnewline
6409 }
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
6410 \renewcommand{\subglossentry}[3]{%
6411   &
6412   \glssubentryitem{##2}%
6413   \glstarget{##2}{\strut}\glossentrydesc{##2} &
6414   ##3\tabularnewline
6415 }
```

Blank row between groups:

```
6416 \renewcommand*{\glsgroupskip}{%
6417   \ifglsgroupskip\else & &\tabularnewline\fi}%
6418 }
```

long3colborder The long3colborder style is like the long3col style but with a border:

```
6419 \newglossarystyle{long3colborder}{%
```

Base it on the glostylelong3col style:

```
6420 \setglossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
6421 \renewenvironment{theglossary}%
6422   {\begin{longtable}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}}%
6423   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
6424 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6425 }
```

long3colheader The long3colheader style is like long3col but with a header row:

```
6426 \newglossarystyle{long3colheader}{%
```

Base it on the glostylelong3col style:

```
6427 \setglossarystyle{long3col}%
```

Set the table's header:

```
6428 \renewcommand*{\glossaryheader}{%
6429   \bfseries\entryname&\bfseries\descriptionname&
6430   \bfseries\pagelistname\tabularnewline\endhead}%
6431 }
```

long3colheaderborder The long3colheaderborder style is like the above but with a border

```
6432 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
6433 \setglossarystyle{long3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
6434 \renewcommand*{\glossaryheader}{{%  
6435 \hline  
6436 \bfseries\entryname&\bfseries\descriptionname&  
6437 \bfseries\pagelistname\tabularnewline\hline\endhead  
6438 \hline\endfoot}%  
6439 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
6440 \newglossarystyle{long4col}{%
```

Use a `longtable` with 4 columns:

```
6441 \renewenvironment{theglossary}{%  
6442 {\begin{longtable}{llll}}%  
6443 {\end{longtable}}%
```

No table header:

```
6444 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6445 \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):

```
6446 \renewcommand{\glossentry}[2]{%  
6447 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
6448 \glossentrydesc{##1} &  
6449 \glossentrysymbol{##1} &  
6450 ##2\tabularnewline  
6451 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
6452 \renewcommand{\subglossentry}[3]{%  
6453 &  
6454 \glssubentryitem{##2}%  
6455 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
6456 \glossentrysymbol{##2} & ##3\tabularnewline  
6457 }%
```

Blank row between groups:

```
6458 \renewcommand*{\glsgroupskip}{%  
6459 \ifglsgnোগroupskip\else & & \tabularnewline\fi}%  
6460 }
```

`long4colheader` The `long4colheader` style is like `long4col` but with a header row.

```
6461 \newglossarystyle{long4colheader}{%
```

Base it on the `glostylelong4col` style:

```
6462 \setglossarystyle{long4col}%
```

Table has a header:

```
6463 \renewcommand*{\glossaryheader}{%
6464   \bfseries\entryname&\bfseries\descriptionname&
6465   \bfseries \symbolname&
6466   \bfseries\pagelistname\tabularnewline\endhead}%
6467 }
```

`long4colborder` The `long4colborder` style is like `long4col` but with a border.

```
6468 \newglossarystyle{long4colborder}{%
```

Base it on the `glostylelong4col` style:

```
6469 \setglossarystyle{long4col}%
```

Use a `longtable` with 4 columns surrounded by vertical lines:

```
6470 \renewenvironment{theglossary}%
6471   {\begin{longtable}{|l|l|l|l|}}%
6472   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
6473 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6474 }
```

`long4colheaderborder` The `long4colheaderborder` style is like the above but with a border.

```
6475 \newglossarystyle{long4colheaderborder}{%
```

Base it on the `glostylelong4col` style:

```
6476 \setglossarystyle{long4col}%
```

Use a `longtable` with 4 columns surrounded by vertical lines:

```
6477 \renewenvironment{theglossary}%
6478   {\begin{longtable}{|l|l|l|l|}}%
6479   {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
6480 \renewcommand*{\glossaryheader}{%
6481   \hline\bfseries\entryname&\bfseries\descriptionname&
6482   \bfseries \symbolname&
6483   \bfseries\pagelistname\tabularnewline\hline\endhead
6484   \hline\endfoot}%
6485 }
```

`altlong4col` The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
6486 \newglossarystyle{altlong4col}{%
```

Base it on the `glostylelong4col` style:

```
6487 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6488 \renewenvironment{theglossary}%
6489   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
6490   {\end{longtable}}%
6491 }
```

altlong4colheader The altlong4colheader style is like altlong4col but with a header row.

```
6492 \newglossarystyle{altlong4colheader}{%
```

Base it on the glostylelong4colheader style:

```
6493 \setglossarystyle{long4colheader}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6494 \renewenvironment{theglossary}%
6495   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
6496   {\end{longtable}}%
6497 }
```

altlong4colborder The altlong4colborder style is like altlong4col but with a border.

```
6498 \newglossarystyle{altlong4colborder}{%
```

Base it on the glostylelong4colborder style:

```
6499 \setglossarystyle{long4colborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6500 \renewenvironment{theglossary}%
6501   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%
6502   {\end{longtable}}%
6503 }
```

ong4colheaderborder The altlong4colheaderborder style is like the above but with a header as well as a border.

```
6504 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the glostylelong4colheaderborder style:

```
6505 \setglossarystyle{long4colheaderborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6506 \renewenvironment{theglossary}%
6507   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%
6508   {\end{longtable}}%
6509 }
```

4.5 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.

```
6510 \ProvidesPackage{glossary-longragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
6511 \RequirePackage{array}
```

Requires the package:

```
6512 \RequirePackage{longtable}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```
6513 \@ifundefined{glsdescwidth}{%
6514   \newlength{glsdescwidth
6515   \setlength{glsdescwidth}{0.6\hsize}
6516 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
6517 \@ifundefined{glspagelistwidth}{%
6518   \newlength{glspagelistwidth
6519   \setlength{glspagelistwidth}{0.1\hsize}
6520 }{}
```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

```
6521 \newglossarystyle{longragged}{%
```

Use longtable with two columns:

```
6522   \renewenvironment{theglossary}{%
6523     {\begin{longtable}{l>{\raggedright}p{glsdescwidth}}}%
6524     {\end{longtable}}}%
```

Do nothing at the start of the environment:

```
6525   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
6526   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
6527   \renewcommand{\glossentry}[2]{%
6528     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6529     \glossentrydesc{##1}\glspostdescription\space ##2%
6530     \tabularnewline
6531   }%
```

Sub entries displayed on the following row without the name:

```

6532 \renewcommand{\subglossentry}[3]{%
6533     &
6534     \glssubentryitem{##2}%
6535     \glstarget{##2}{\strut}\glossentrydesc{##2}%
6536     \glspostdescription\space ##3%
6537     \tabularnewline
6538 }%

```

Blank row between groups:

```

6539 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
6540 }

```

`longraggedborder` The `longraggedborder` style is like the above, but with horizontal and vertical lines:

6541 \newglossarystyle{longraggedborder}{%

Base it on the `glostylelongragged` style:

```
6542 \setglossarystyle{longragged}%
```

Use longtable with two columns with vertical lines between each column:

```

6543 \renewenvironment{theglossary}{%
6544 \begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|}%
6545 {\end{longtable}}%

```

Place horizontal lines at the head and foot of the table:

```
6546 \renewcommand*{\glossaryheader}{\hrline\endhead\hrline\endfoot}%
6547 }
```

`longraggedheader` The `longraggedheader` style is like the `longragged` style but with a header:

6548 \newglossarystyle{longraggedheader}{%

Base it on the `glostylelongragged` style:

```
6549 \setglossarystyle{longragged}%
```

Set the table's header:

```

6550 \renewcommand*{\glossaryheader}{%
6551 \bfseries \entryname & \bfseries \descriptionname
6552 \tabularnewline\endhead}%
6553 }

```

`longraggedheaderborder` The `longraggedheaderborder` style is like the `longragged` style but with a header and border:

```
6554 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
6555 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```

6556 \renewcommand*{\glossaryheader}{%
6557 \hline\bfseries \entryname & \bfseries \descriptionname

```



```

6558 \tabularnewline\hline
6559 \endhead
6560 \hline\endfoot}%
6561 }

```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```

6562 \newglossarystyle{longragged3col}{%

```

Use a longtable with 3 columns:

```

6563 \renewenvironment{theglossary}%
6564 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
6565 >{\raggedright}p{\glspagelistwidth}}}%
6566 {\end{longtable}}%

```

No table header:

```

6567 \renewcommand*{\glossaryheader}{}%

```

No headings between groups:

```

6568 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

6569 \renewcommand{\glossentry}[2]{%
6570 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6571 \glossentrydesc{##1} & ##2\tabularnewline
6572 }%

```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```

6573 \renewcommand{\subglossentry}[3]{%
6574 &
6575 \glssubentryitem{##2}%
6576 \glstarget{##2}{\strut}\glossentrydesc{##2} &
6577 ##3\tabularnewline
6578 }%

```

Blank row between groups:

```

6579 \renewcommand*{\glsgroupskip}{%
6580 \ifglsnogroupskip\else & &\tabularnewline\fi}%
6581 }

```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```

6582 \newglossarystyle{longragged3colborder}{%

```

Base it on the `glostylelongragged3col` style:

```

6583 \setglossarystyle{longragged3col}%

```

Use a longtable with 3 columns with vertical lines around them:

```

6584 \renewenvironment{theglossary}%
6585 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
6586 >{\raggedright}p{\glspagelistwidth}|}%
6587 {\end{longtable}}%

```

Place horizontal lines at the head and foot of the table:

```
6588 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6589 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
6590 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
6591 \setglossarystyle{longragged3col}%
```

Set the table's header:

```
6592 \renewcommand*{\glossaryheader}{%
6593 \bfseries\entryname&\bfseries\descriptionname&
6594 \bfseries\pagelistname\tabularnewline\endhead}%
6595 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
6596 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
6597 \setglossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
6598 \renewcommand*{\glossaryheader}{%
6599 \hline
6600 \bfseries\entryname&\bfseries\descriptionname&
6601 \bfseries\pagelistname\tabularnewline\hline\endhead
6602 \hline\endfoot}%
6603 }
```

`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.

```
6604 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6605 \renewenvironment{theglossary}%
6606 {\begin{longtable}{l>{\raggedright}p{\glstdescwidth}l%
6607 >{\raggedright}p{\glspagelistwidth}}}%
6608 {\end{longtable}}%
```

No table header:

```
6609 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6610 \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):

```
6611 \renewcommand{\glossentry}[2]{}%
```

```

6612 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6613 \glossentrydesc{##1} & \glossentrydesc{##1} &
6614 ##2\tabularnewline
6615 }%

```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```

6616 \renewcommand{\subglossentry}[3]{%
6617   &
6618   \glssubentryitem{##2}%
6619   \glstarget{##2}{\strut}\glossentrydesc{##2} &
6620   \glossentrysymbol{##2} & ##3\tabularnewline
6621 }%

```

Blank row between groups:

```

6622 \renewcommand*\glsgroupskip{%
6623   \ifglsgroupskip\else & & \tabularnewline\fi}%
6624 }

```

`longragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```

6625 \newglossarystyle{altlongragged4colheader}{%

```

Base it on the `glostylealtlongragged4col` style:

```

6626 \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

6627 \renewenvironment{theglossary}%
6628   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
6629     >{\raggedright}p{\glspagelistwidth}}}%
6630   {\end{longtable}}%

```

Table has a header:

```

6631 \renewcommand*\glossaryheader{%
6632   \bfseries\entryname&\bfseries\descriptionname&
6633   \bfseries \symbolname&
6634   \bfseries\pagelistname\tabularnewline\endhead}%
6635 }

```

`longragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```

6636 \newglossarystyle{altlongragged4colborder}{%

```

Base it on the `glostylealtlongragged4col` style:

```

6637 \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

6638 \renewenvironment{theglossary}%
6639   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|%
6640     >{\raggedright}p{\glspagelistwidth}|}%
6641   {\end{longtable}}%

```

Add horizontal lines to the head and foot of the table:

```
6642 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6643 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
6644 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
6645 \setglossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
6646 \renewenvironment{theglossary}%
6647 {\begin{longtable}{|l|>{\raggedright}p{\glsgdescwidth}|l|}%
6648 >{\raggedright}p{\glspagelistwidth}|}%
6649 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
6650 \renewcommand*{\glossaryheader}{%
6651 \hline\bfseries\entryname&\bfseries\descriptionname&
6652 \bfseries \symbolname&
6653 \bfseries\pagelistname\tabularnewline\hline\endhead
6654 \hline\endfoot}%
6655 }
```

4.6 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.

```
6656 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```
6657 \RequirePackage{multicol}
6658 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
6659 \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.)

```
6660 \newglossarystyle{mcolindex}{%
6661 \setglossarystyle{index}%
6662 \renewenvironment{theglossary}%
6663 {%
```

```

6664 \begin{multicols}{\glsmcols}
6665 \setlength{\parindent}{0pt}%
6666 \setlength{\parskip}{0pt plus 0.3pt}%
6667 \let\item\@idxitem}%
6668 {\end{multicols}}%
6669 }

```

mcolindexgroup As mcolindex but has headings:

```

6670 \newglossarystyle{mcolindexgroup}{%
6671 \setglossarystyle{mcolindex}%
6672 \renewcommand*{\glsgroupheading}[1]{%
6673 \item\textbf{\glsgroupheading{##1}}\indexspace}%
6674 }

```

mcolindexhypergroup The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

6675 \newglossarystyle{mcolindexhypergroup}{%
  Base it on the glostylemcolindex style:
6676 \setglossarystyle{mcolindex}%
  Put navigation links to the groups at the start of the glossary:
6677 \renewcommand*{\glossaryheader}{%
6678 \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.
6679 \renewcommand*{\glsgroupheading}[1]{%
6680 \item\textbf{\glshnavhypertarget{##1}}{\glsgroupheading{##1}}}%
6681 \indexspace}%
6682 }

```

mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

6683 \newglossarystyle{mcoltree}{%
6684 \setglossarystyle{tree}%
6685 \renewenvironment{theglossary}%
6686 {%
6687 \begin{multicols}{\glsmcols}
6688 \setlength{\parindent}{0pt}%
6689 \setlength{\parskip}{0pt plus 0.3pt}%
6690 }%
6691 {\end{multicols}}%
6692 }

```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```

6693 \newglossarystyle{mcoltreegroup}{%
  Base it on the glostylemcoltree style:
6694 \setglossarystyle{mcoltree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```
6695 \renewcommand{\glsgroupheading}[1]{\par
6696 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
6697 }
```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
6698 \newglossarystyle{mcoltreehypergroup}{%
```

Base it on the glostylemcoltree style:

```
6699 \setglossarystyle{mcoltree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
6700 \renewcommand*{\glossaryheader}{%
6701 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
6702 \renewcommand*{\glsgroupheading}[1]{%
6703 \par\noindent
6704 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
6705 \indexspace}%
6706 }
```

mcoltreenoname Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```
6707 \newglossarystyle{mcoltreenoname}{%
6708 \setglossarystyle{treenoname}%
6709 \renewenvironment{theglossary}%
6710 {%
6711 \begin{multicols}{\glsmcols}
6712 \setlength{\parindent}{0pt}%
6713 \setlength{\parskip}{0pt plus 0.3pt}%
6714 }%
6715 {\end{multicols}}}%
6716 }
```

mcoltreenonamegroup Like the mcoltreenoname style but the glossary groups have headings.

```
6717 \newglossarystyle{mcoltreenonamegroup}{%
```

Base it on the glostylemcoltreenoname style:

```
6718 \setglossarystyle{mcoltreenoname}%
```

Give each group a heading:

```
6719 \renewcommand{\glsgroupheading}[1]{\par
6720 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
6721 }
```

treenonamehypergroup The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```
6722 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreename` style:

```
6723 \setglossarystyle{mcoltreename}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
6724 \renewcommand*{\glossaryheader}{%
```

```
6725 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
6726 \renewcommand*{\glsgroupheading}[1]{%
```

```
6727 \par\noindent
```

```
6728 \textbf{\glsnavigationhypertarget{##1}{\glsgroupheading{##1}}}\par
```

```
6729 \indexspace}%
```

```
6730 }
```

`mcolalttree` Multi-column index style. Same as the `alttree`, but puts the glossary in multiple columns.

```
6731 \newglossarystyle{mcolalttree}{%
```

```
6732 \setglossarystyle{alttree}%
```

```
6733 \renewenvironment{theglossary}%
```

```
6734 {%
```

```
6735 \begin{multicols}{\glsmcols}
```

```
6736 \def\@gls@prevlevel{-1}%
```

```
6737 \mbox{}\par
```

```
6738 }%
```

```
6739 {\par\end{multicols}}%
```

```
6740 }
```

`mcolalttreegroup` Like the `mcolalttree` style but the glossary groups have headings.

```
6741 \newglossarystyle{mcolalttreegroup}{%
```

Base it on the `glostylemcolalttree` style:

```
6742 \setglossarystyle{mcolalttree}%
```

Give each group a heading.

```
6743 \renewcommand{\glsgroupheading}[1]{\par
```

```
6744 \def\@gls@prevlevel{-1}%
```

```
6745 \hangindent0pt\relax
```

```
6746 \parindent0pt\relax
```

```
6747 \textbf{\glsgroupheading{##1}}\par\indexspace}%
```

```
6748 }
```

`olalttreehypergroup` The `mcolalttreehypergroup` style is like the `mcolalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
6749 \newglossarystyle{olalttreehypergroup}{%
```

Base it on the `glostylemcolalttree` style:

```
6750 \setglossarystyle{mcolalttree}%
```

Put the navigation links in the header

```
6751 \renewcommand*{\glossaryheader}{%
6752   \par
6753   \def\@gls@prevlevel{-1}%
6754   \hangindent0pt\relax
6755   \parindent0pt\relax
6756   \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
6757 \renewcommand*{\glsgroupheading}[1]{%
6758   \par
6759   \def\@gls@prevlevel{-1}%
6760   \hangindent0pt\relax
6761   \parindent0pt\relax
6762   \textbf{\glsnavhypertarget{##1}\glsgetgrouptitle{##1}}\par
6763   \indexspace}}
```

4.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
6764 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
6765 \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.

```
6766 \@ifundefined{glsdescwidth}{%
6767   \newlength{glsdescwidth
6768   \setlength{glsdescwidth}{0.6\hsize}
6769 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
6770 \@ifundefined{glspagelistwidth}{%
6771   \newlength{glspagelistwidth
6772   \setlength{glspagelistwidth}{0.1\hsize}
6773 }{}
```

`super` The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
6774 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
6775   \renewenvironment{theglossary}%
6776   {\tablehead{}\tabletail}%
6777   \begin{supertabular}{lp{glsdescwidth}}%
6778   {\end{supertabular}}%
```


Do nothing at the start of the table:

```
6779 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6780 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
6781 \renewcommand{\glossentry}[2]{%
6782   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6783   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
6784 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
6785 \renewcommand{\subglossentry}[3]{%
6786   &
6787   \glssubentryitem{##2}%
6788   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
6789   ##3\tabularnewline
6790 }%
```

Blank row between groups:

```
6791 \renewcommand*{\glsgroupskip}{}%
6792 \ifglsgnognoskip\else & \tabularnewline\fi}%
6793 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
6794 \newglossarystyle{superborder}{}%
```

Base it on the `glostylesuper` style:

```
6795 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns and a horizontal line in the head and tail:

```
6796 \renewenvironment{theglossary}%
6797   {\tablehead{\hline}\tabletail{\hline}%
6798   \begin{supertabular}{|l|p{\glsdescwidth}|}%
6799   {\end{supertabular}}%
6800 }
```

superheader The superheader style is like the super style, but with a header:

```
6801 \newglossarystyle{superheader}{}%
```

Base it on the `glostylesuper` style:

```
6802 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
6803 \renewenvironment{theglossary}%
6804   {\tablehead{\bfseries \entryname &
6805     \bfseries\descriptionname\tabularnewline}%

```

```

6806 \tabletail{ }%
6807 \begin{supertabular}{lp{\glsdescwidth}}}%
6808 {\end{supertabular}}}%
6809 }

```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```

6810 \newglossarystyle{superheaderborder}{%
    Base it on the glostylesuper style:
6811 \setglossarystyle{super}%
    Put the glossary in a supertabular environment with two columns, a header and
    horizontal lines above and below the table:
6812 \renewenvironment{theglossary}%
6813 {\tablehead{\hline\bfseries \entryname &
6814 \bfseries \descriptionname\tabularnewline\hline}%
6815 \tabletail{\hline}
6816 \begin{supertabular}{|lp{\glsdescwidth}|}%
6817 {\end{supertabular}}}%
6818 }

```

super3col The super3col style is like the super style, but with 3 columns:

```

6819 \newglossarystyle{super3col}{%
    Put the glossary in a supertabular environment with three columns and no head
    or tail:
6820 \renewenvironment{theglossary}%
6821 {\tablehead{} \tabletail{ }%
6822 \begin{supertabular}{lp{\glsdescwidth}p{\glspagerlistwidth}}}%
6823 {\end{supertabular}}}%
    Do nothing at the start of the table:
6824 \renewcommand*{\glossaryheader}{ }%
    No group headings:
6825 \renewcommand*{\glsgroupheading}[1]{ }%
    Main (level 0) entries on a row (name in first column, description in second
    column, page list in last column):
6826 \renewcommand{\glossentry}[2]{%
6827 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6828 \glossentrydesc{##1} & ##2\tabularnewline
6829 }%
    Sub entries on a row (no name, description in second column, page list in last
    column):
6830 \renewcommand{\subglossentry}[3]{%
6831 &
6832 \glssubentryitem{##2}%
6833 \glstarget{##2}{\strut}\glossentrydesc{##2} &
6834 ##3\tabularnewline
6835 }%

```

Blank row between groups:

```
6836 \renewcommand*{\glsgroupskip}{%
6837 \ifglsgroupskip\else & &\tabularnewline\fi}%
6838 }
```

super3colborder The `super3colborder` style is like the `super3col` style, but with a border:

```
6839 \newglossarystyle{super3colborder}{%
Base it on the glostylesuper3col style:
6840 \setglossarystyle{super3col}%
Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
6841 \renewenvironment{theglossary}%
6842 {\tablehead{\hline}\tabletail{\hline}%
6843 \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
6844 {\end{supertabular}}%
6845 }
```

super3colheader The `super3colheader` style is like the `super3col` style but with a header row:

```
6846 \newglossarystyle{super3colheader}{%
Base it on the glostylesuper3col style:
6847 \setglossarystyle{super3col}%
Put the glossary in a supertabular environment with three columns, a header and no tail:
6848 \renewenvironment{theglossary}%
6849 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
6850 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
6851 \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
6852 {\end{supertabular}}%
6853 }
```

super3colheaderborder The `super3colheaderborder` style is like the `super3col` style but with a header and border:

```
6854 \newglossarystyle{super3colheaderborder}{%
Base it on the glostylesuper3colborder style:
6855 \setglossarystyle{super3colborder}%
Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:
6856 \renewenvironment{theglossary}%
6857 {\tablehead{\hline
6858 \bfseries\entryname&\bfseries\descriptionname&
6859 \bfseries\pagelistname\tabularnewline\hline}%
6860 \tabletail{\hline}%
6861 \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
6862 {\end{supertabular}}%
6863 }
```

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.

```
6864 \newglossarystyle{super4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
6865 \renewenvironment{theglossary}%
6866 {\tablehead{}\tabletail}%
6867 \begin{supertabular}{1111}}{%
6868 \end{supertabular}}%
```

Do nothing at the start of the table:

```
6869 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6870 \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:

```
6871 \renewcommand{\glossentry}[2]{%
6872 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6873 \glossentrydesc{##1} &
6874 \glossentrysymbol{##1} & ##3\tabularnewline
6875 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
6876 \renewcommand{\subglossentry}[3]{%
6877 &
6878 \glssubentryitem{##2}%
6879 \glstarget{##2}{\strut}\glossentrydesc{##2} &
6880 \glossentrysymbol{##2} & ##3\tabularnewline
6881 }%
```

Blank row between groups:

```
6882 \renewcommand*{\glsgroupskip}{%
6883 \ifglsgroupskip\else & & \tabularnewline\fi}%
6884 }
```

super4colheader The **super4colheader** style is like the **super4col** but with a header row.

```
6885 \newglossarystyle{super4colheader}{%
```

Base it on the **glostylessuper4col** style:

```
6886 \setglossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```
6887 \renewenvironment{theglossary}%
6888 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
6889 \bfseries\symbolname &
6890 \bfseries\pagelistname\tabularnewline}}%
```

```

6891     \tabletail{}}%
6892     \begin{supertabular}{|l|l|l|l|}%
6893     {\end{supertabular}}}%
6894 }

```

super4colborder The `super4colborder` style is like the `super4col` but with a border.

```

6895 \newglossarystyle{super4colborder}{%
    Base it on the glostylesuper4col style:
6896   \setglossarystyle{super4col}%
    Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:
6897   \renewenvironment{theglossary}%
6898     {\tablehead{\hline}\tabletail{\hline}}%
6899     \begin{supertabular}{|l|l|l|l|}%
6900     {\end{supertabular}}}%
6901 }

```

super4colheaderborder The `super4colheaderborder` style is like the `super4col` but with a header and border.

```

6902 \newglossarystyle{super4colheaderborder}{%
    Base it on the glostylesuper4col style:
6903   \setglossarystyle{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:
6904   \renewenvironment{theglossary}%
6905     {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
6906               \bfseries\symbolname &
6907               \bfseries\pagelistname\tabularnewline\hline}%
6908     \tabletail{\hline}%
6909     \begin{supertabular}{|l|l|l|l|}%
6910     {\end{supertabular}}}%
6911 }

```

altsuper4col The `altsuper4col` glossary style is like `super4col` but has provision for multiline descriptions.

```

6912 \newglossarystyle{altsuper4col}{%
    Base it on the glostylesuper4col style:
6913   \setglossarystyle{super4col}%
    Put the glossary in a supertabular environment with four columns and no head or tail:
6914   \renewenvironment{theglossary}%
6915     {\tablehead{}\tabletail{}}%
6916     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
6917     {\end{supertabular}}}%
6918 }

```

altsuper4colheader The altsuper4colheader style is like the altsuper4col but with a header row.

```

6919 \newglossarystyle{altsuper4colheader}{%
    Base it on the glostylesuper4colheader style:
6920   \setglossarystyle{super4colheader}%

    Put the glossary in a supertabular environment with four columns, a header and
    no tail:
6921   \renewenvironment{theglossary}%
6922     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
6923       \bfseries\symbolname &
6924       \bfseries\pagelistname\tabularnewline}\tabletail{}}%
6925     \begin{supertabular}{\lp{\glsgdescwidth}\lp{\glspagelistwidth}}}%
6926     {\end{supertabular}}}%
6927 }
```

altsuper4colborder The altsuper4colborder style is like the altsuper4col but with a border.

```

6928 \newglossarystyle{altsuper4colborder}{%
    Base it on the glostylesuper4colborder style:
6929   \setglossarystyle{super4colborder}%

    Put the glossary in a supertabular environment with four columns and a horizontal
    line in the head and tail:
6930   \renewenvironment{theglossary}%
6931     {\tablehead{\hline}\tabletail{\hline}%
6932     \begin{supertabular}%
6933       {\lllp{\glsgdescwidth}\lllp{\glspagelistwidth}}}%
6934     {\end{supertabular}}}%
6935 }
```

per4colheaderborder The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```

6936 \newglossarystyle{altsuper4colheaderborder}{%
    Base it on the glostylesuper4colheaderborder style:
6937   \setglossarystyle{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:
6938   \renewenvironment{theglossary}%
6939     {\tablehead{\hline
6940       \bfseries\entryname &
6941       \bfseries\descriptionname &
6942       \bfseries\symbolname &
6943       \bfseries\pagelistname\tabularnewline\hline}%
6944     \tabletail{\hline}%
6945     \begin{supertabular}%
6946       {\lllp{\glsgdescwidth}\lllp{\glspagelistwidth}}}%
6947     {\end{supertabular}}}%
6948 }
```

4.8 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.

6949 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]

Requires the package:

6950 \RequirePackage{array}

Requires the package:

6951 \RequirePackage{supertabular}

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

6952 \@ifundefined{glsdescwidth}{%

6953 \newlength{glsdescwidth}

6954 \setlength{glsdescwidth}{0.6\hsize}

6955 }{}

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

6956 \@ifundefined{glspagelistwidth}{%

6957 \newlength{glspagelistwidth}

6958 \setlength{glspagelistwidth}{0.1\hsize}

6959 }{}

`superragged` The superragged glossary style uses the supertabular environment.

6960 \newglossarystyle{superragged}{%

Put the glossary in a supertabular environment with two columns and no head or tail:

6961 \renewenvironment{theglossary}{%

6962 {\tablehead{}\tabletail}{%

6963 \begin{supertabular}{1>{\raggedright}p{glsdescwidth}}{%

6964 {\end{supertabular}}{%

Do nothing at the start of the table:

6965 \renewcommand*{\glossaryheader}{}{%

No group headings:

6966 \renewcommand*{\glsgroupheading}[1]{}{%

Main (level 0) entries put in a row (name in first column, description and page list in second column):

6967 \renewcommand{\glossentry}[2]{%

6968 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &

6969 \glossentrydesc{##1}\glspostdescription\space ##2%

6970 \tabularnewline

6971 }{%

Sub entries put in a row (no name, description and page list in second column):

```

6972 \renewcommand{\subglossentry}[3]{%
6973     &
6974     \glssubentryitem{##2}%
6975     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
6976     ##3%
6977     \tabularnewline
6978 }%
```

Blank row between groups:

```

6979 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
6980 }
```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```

6981 \newglossarystyle{superraggedborder}{%
    Base it on the glostylesuperragged style:
6982 \setglossarystyle{superragged}%
    Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:
6983 \renewenvironment{theglossary}%
6984     {\tablehead{\hline}\tabletail{\hline}%
6985     \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|}}%
6986     {\end{supertabular}}%
6987 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```

6988 \newglossarystyle{superraggedheader}{%
    Base it on the glostylesuperragged style:
6989 \setglossarystyle{superragged}%
    Put the glossary in a supertabular environment with two columns, a header and no tail:
6990 \renewenvironment{theglossary}%
6991     {\tablehead{\bfseries \entryname & \bfseries \descriptionname
6992         \tabularnewline}%
6993     \tabletail{}%
6994     \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}}}%
6995     {\end{supertabular}}%
6996 }
```

rraggedheaderborder The superraggedheaderborder style is like the superragged style but with a header and border:

```

6997 \newglossarystyle{superraggedheaderborder}{%
    Base it on the glostylesuper style:
6998 \setglossarystyle{superragged}%
```


Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```
6999 \renewenvironment{theglossary}%
7000   {\tablehead{\hline\bfseries \entryname &
7001     \bfseries \descriptionname\tabularnewline\hline}%
7002   \tabletail{\hline}
7003   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}}%
7004   {\end{supertabular}}%
7005 }
```

superragged3col The superragged3col style is like the superragged style, but with 3 columns:

```
7006 \newglossarystyle{superragged3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```
7007 \renewenvironment{theglossary}%
7008   {\tablehead{}\tabletail}%
7009   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}%
7010     >{\raggedright}p{\glspagelistwidth}|}}%
7011   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7012 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7013 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7014 \renewcommand{\glossentry}[2] {%
7015   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7016   \glossentrydesc{##1} &
7017   ##2\tabularnewline
7018 }%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
7019 \renewcommand{\subglossentry}[3] {%
7020   &
7021   \glssubentryitem{##2}%
7022   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7023   ##3\tabularnewline
7024 }%
```

Blank row between groups:

```
7025 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & &\tabularnewline\fi}%
7026 }
```

superragged3colborder The superragged3colborder style is like the superragged3col style, but with a border:

```
7027 \newglossarystyle{superragged3colborder}{%
```

Base it on the `glostylesuperragged3col` style:

```
7028 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```
7029 \renewenvironment{theglossary}%  
7030 {\tablehead{\hline}\tabletail{\hline}%  
7031 \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}|%  
7032 >\raggedright}p{\glspagelistwidth}|}%  
7033 {\end{supertabular}}%  
7034 }
```

`superragged3colheader` The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```
7035 \newglossarystyle{superragged3colheader}{%
```

Base it on the `glostylesuperragged3col` style:

```
7036 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```
7037 \renewenvironment{theglossary}%  
7038 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
7039 \bfseries\pagelistname\tablearnewline}\tabletail{}}%  
7040 \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}%  
7041 >\raggedright}p{\glspagelistwidth}}}%  
7042 {\end{supertabular}}%  
7043 }
```

`superragged3colheaderborder` The `superragged3colheaderborder` style is like the `superragged3col` style but with a header and border:

```
7044 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the `glostylesuperragged3colborder` style:

```
7045 \setglossarystyle{superragged3colborder}%
```

Put the glossary in a `supertabular` environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
7046 \renewenvironment{theglossary}%  
7047 {\tablehead{\hline  
7048 \bfseries\entryname&\bfseries\descriptionname&  
7049 \bfseries\pagelistname\tablearnewline\hline}%  
7050 \tabletail{\hline}%  
7051 \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}|%  
7052 >\raggedright}p{\glspagelistwidth}|}%  
7053 {\end{supertabular}}%  
7054 }
```

`altsuperragged4col` The `altsuperragged4col` glossary style is like `altsuper4col` style in the package but uses ragged right formatting in the description and page list columns.

```
7055 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
7056 \renewenvironment{theglossary}%
7057   {\tablehead{}\tabletail{}}%
7058   \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}l%
7059     >{\raggedright}p{\glspagelistwidth}}}%
7060   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7061 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7062 \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:

```
7063 \renewcommand{\glossentry}[2]{%
7064   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7065   \glossentrydesc{##1} &
7066   \glossentrysymbol{##1} & ##2\tabularnewline
7067 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
7068 \renewcommand{\subglossentry}[3]{%
7069   &
7070   \glssubentryitem{##2}%
7071   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7072   \glossentrysymbol{##2} & ##3\tabularnewline
7073 }%
```

Blank row between groups:

```
7074 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & & \tabularnewline\fi}%
7075 }
```

`\perragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```
7076 \newglossarystyle{altsuperragged4colheader}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
7077 \setglossarystyle{altsuperragged4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```
7078 \renewenvironment{theglossary}%
7079   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7080     \bfseries\symbolname &
7081     \bfseries\pagelistname\tabletabularnewline}\tabletail{}}%
7082   \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}l%
7083     >{\raggedright}p{\glspagelistwidth}}}%
7084   {\end{supertabular}}%
7085 }
```

`altsuperragged4colborder` The `altsuperragged4colborder` style is like the `altsuperragged4col` style but with a border.

```
7086 \newglossarystyle{altsuperragged4colborder}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
7087 \setglossarystyle{altsuper4col}{%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
7088 \renewenvironment{theglossary}{%
7089 {\tablehead{\hline}\tabletail{\hline}%
7090 \begin{supertabular}%
7091 {\|l|>\raggedright}p{\glsgdescwidth}|l|}%
7092 >\raggedright}p{\glspagelistwidth}|}}%
7093 {\end{supertabular}}}%
7094 }
```

`altd4colheaderborder` The `altd4colheaderborder` style is like the `altd4col` style but with a header and border.

```
7095 \newglossarystyle{altd4colheaderborder}{%
```

Base it on the `glostylealtd4col` style:

```
7096 \setglossarystyle{altd4col}{%
```

Put the glossary in a `supertabular` environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:

```
7097 \renewenvironment{theglossary}{%
7098 {\tablehead{\hline
7099 \bfseries\entryname &
7100 \bfseries\descriptionname &
7101 \bfseries\symbolname &
7102 \bfseries\pagelistname\tabularnewline\hline}%
7103 \tabletail{\hline}%
7104 \begin{supertabular}%
7105 {\|l|>\raggedright}p{\glsgdescwidth}|l|}%
7106 >\raggedright}p{\glspagelistwidth}|}}%
7107 {\end{supertabular}}}%
7108 }
```

4.9 Tree Styles (`glossary-tree.sty`)

The style file defines glossary styles that have a tree-like structure. These are designed for hierarchical glossaries.

```
7109 \ProvidesPackage{glossary-tree}[2013/11/14 v4.0 (NLCT)]
```

`index` The index glossary style is similar in style to the way indices are usually typeset using `\item`, `\subitem` and `\subsubitem`. The entry name is set in bold. If an entry has a symbol, it is placed in brackets after the name. Then the description is displayed, followed by the number list. This style allows up to three levels.

```
7110 \newglossarystyle{index}{%
```

Set the paragraph indentation and skip and define `\item` to be the same as that used by theindex:

```
7111 \renewenvironment{theglossary}%
7112   {\setlength{\parindent}{0pt}%
7113    \setlength{\parskip}{0pt plus 0.3pt}%
7114    \let\item\@idxitem}%
7115   {\par}%
```

Do nothing at the start of the environment:

```
7116 \renewcommand*{\glossaryheader}{}%
```

No group headers:

```
7117 \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.

```
7118 \renewcommand*{\glossentry}[2]{%
7119   \item\glstentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7120   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7121   \space \glossentrydesc{##1}\glspostdescription\space ##2%
7122 }%
```

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 `\glossentry`, 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.

```
7123 \renewcommand{\subglossentry}[3]{%
7124   \ifcase##1\relax
7125     % level 0
7126     \item
7127   \or
7128     % level 1
7129     \subitem
7130     \glssubentryitem{##2}%
7131   \else
7132     % all other levels
7133     \subsubitem
7134   \fi
7135   \textbf{\glstarget{##2}{\glossentryname{##2}}}%
7136   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
7137   \space\glossentrydesc{##2}\glspostdescription\space ##3%
7138 }%
```

Vertical gap between groups is the same as that used by indices:

```
7139 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

indexgroup The `indexgroup` style is like the `index` style but has headings.

```
7140 \newglossarystyle{indexgroup}{%
```

Base it on the `glostyleindex` style:

```
7141 \setglossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```
7142 \renewcommand*{\glsgroupheading}[1]{%
7143   \item\textbf{\glsgetgrouptitle{##1}}\indexspace}%
7144 }
```

`indexhypergroup` The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```
7145 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```
7146 \setglossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```
7147 \renewcommand*{\glossaryheader}{%
7148   \item\textbf{\glsnavigation}\indexspace}%
```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```
7149 \renewcommand*{\glsgroupheading}[1]{%
7150   \item\textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
7151   \indexspace}%
7152 }
```

`tree` The `tree` glossary style is similar in style to the `index` style, but can have arbitrary levels.

```
7153 \newglossarystyle{tree}{%
```

Set the paragraph indentation and skip:

```
7154 \renewenvironment{theglossary}%
7155   {\setlength{\parindent}{0pt}%
7156    \setlength{\parskip}{0pt plus 0.3pt}}%
7157   {}%
```

Do nothing at the start of the `theglossary` environment:

```
7158 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7159 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
7160 \renewcommand{\glossentry}[2]{%
7161   \hangindent0pt\relax
7162   \parindent0pt\relax
7163   \glsentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7164   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7165   \space\glossentrydesc{##1}\glspostdescription\space##2\par
7166 }
```

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.

```

7167 \renewcommand{\subglossentry}[3]{%
7168   \hangindent##1\glstreeindent\relax
7169   \parindent##1\glstreeindent\relax
7170   \ifnum##1=1\relax
7171     \glssubentryitem{##2}%
7172   \fi
7173   \textbf{\glstarget{##2}{\glossentryname{##2}}}%
7174   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
7175   \space\glossentrydesc{##2}\glspostdescription\space ##3\par
7176 }%
```

Vertical gap between groups is the same as that used by indices:

```

7177 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

treegroup Like the tree style but the glossary groups have headings.

```

7178 \newglossarystyle{treegroup}{%
```

Base it on the `glostyletree` style:

```

7179 \setglossarystyle{tree}%
```

Each group has a heading (in bold) followed by a vertical gap):

```

7180 \renewcommand{\glsgroupheading}[1]{\par
7181   \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7182 }
```

treehypergroup The `treehypergroup` style is like the `treegroup` style, but has a set of links to the groups at the start of the glossary.

```

7183 \newglossarystyle{treehypergroup}{%
```

Base it on the `glostyletree` style:

```

7184 \setglossarystyle{tree}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```

7185 \renewcommand*{\glossaryheader}{%
7186   \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```

7187 \renewcommand*{\glsgroupheading}[1]{%
7188   \par\noindent
7189   \textbf{\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}\par
7190   \indexspace}%
7191 }
```

\glstreeindent Length governing left indent for each level of the tree style.

```

7192 \newlength\glstreeindent
7193 \setlength{\glstreeindent}{10pt}
```

treenoname The **treenoname** glossary style is like the **tree** style, but doesn't print the name or symbol for sub-levels.

```
7194 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
7195 \renewenvironment{theglossary}{%
7196   {\setlength{\parindent}{0pt}}%
7197   \setlength{\parskip}{0pt plus 0.3pt}}%
7198   {}}%
```

No header:

```
7199 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7200 \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.

```
7201 \renewcommand{\glossentry}[2]{%
7202   \hangindent0pt\relax
7203   \parindent0pt\relax
7204   \glstryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7205   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7206   \space\glossentrydesc{##1}\glspostdescription\space##2\par
7207 }%
```

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.

```
7208 \renewcommand{\subglossentry}[3]{%
7209   \hangindent##1\glstreeindent\relax
7210   \parindent##1\glstreeindent\relax
7211   \ifnum##1=1\relax
7212     \glssubentryitem{##2}%
7213   \fi
7214   \glstarget{##2}{\strut}%
7215   \glossentrydesc{##2}\glspostdescription\space##3\par
7216 }%
```

Vertical gap between groups is the same as that used by indices:

```
7217 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7218 }
```

treenonamegroup Like the **treenoname** style but the glossary groups have headings.

```
7219 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
7220 \setglossarystyle{treenoname}%
```

Give each group a heading:

```
7221 \renewcommand{\glsgroupheading}[1]{\par
7222   \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7223 }
```


`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
7224 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
7225 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7226 \renewcommand*{\glossaryheader}{%
```

```
7227 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7228 \renewcommand*{\glsgroupheading}[1]{%
```

```
7229 \par\noindent
```

```
7230 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
```

```
7231 \indexspace}%
```

```
7232 }
```

`\glsssetwidest` `\glsssetwidest[<level>]{<text>}` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```
7233 \newcommand*{\glsssetwidest}[2][0]{%
```

```
7234 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
```

```
7235 #2}%
```

```
7236 }
```

`\@glswidestname` Initialise `\@glswidestname`.

```
7237 \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`.

```
7238 \newglossarystyle{alttree}{%
```

Redefine the `theglossary` environment.

```
7239 \renewenvironment{theglossary}%
```

```
7240 {\def\@gls@prevlevel{-1}%
```

```
7241 \mbox{}\par}%
```

```
7242 {\par}%
```

Set the header and group headers to nothing.

```
7243 \renewcommand*{\glossaryheader}{}%
```

```
7244 \renewcommand*{\glsgroupheading}[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
7245 \renewcommand{\glossentry}[2]{%
```

If the level hasn't changed, keep the same settings, otherwise change `\glstreeindent` accordingly.

```
7246 \ifnum\@gls@prevlevel=0\relax
```

```
7247 \else
```

Find out how big the indentation should be by measuring the widest entry.

```
7248      \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
```

Set the hangindent and paragraph indent.

```
7249      \hangindent\glstreeindent
```

```
7250      \parindent\glstreeindent
```

```
7251      \fi
```

Put the name to the left of the paragraph block.

```
7252      \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
```

```
7253        \glstentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
7254      \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
```

Do the description followed by the description terminator and location list.

```
7255      \glossentrydesc{##1}\glspostdescription \space ##2\par
```

Set the previous level to 0.

```
7256      \def\@gls@prevlevel{0}%
```

```
7257    }%
```

Redefine the way sub-entries are displayed.

```
7258      \renewcommand{\subglossentry}[3]{%
```

Increment and display the sub-entry counter if this is a level 1 entry and the sub-entry counter is in use.

```
7259      \ifnum##1=1\relax
```

```
7260        \glssubentryitem{##2}%
```

```
7261      \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```
7262      \ifnum\@gls@prevlevel=##1\relax
```

```
7263      \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in \gls@tmplen

```
7264      \@ifundefined{@glswidestname\romannumeral##1}{%
```

```
7265        \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}}{%
```

```
7266        \settowidth{\gls@tmplen}{\textbf{%
```

```
7267          \csname @glswidestname\romannumeral##1\endcsname\space}}}%
```

Determine if going up or down a level

```
7268      \ifnum\@gls@prevlevel<##1\relax
```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```
7269        \setlength\glstreeindent\gls@tmplen
```

```
7270        \addtolength\glstreeindent\parindent
```

```
7271        \parindent\glstreeindent
```

```
7272      \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`.

```

7273      \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
7274      \settowidth{\glstreeindent}{\textbf{%
7275      \@glswidestname\space}}}{%
7276      \settowidth{\glstreeindent}{\textbf{%
7277      \csname @glswidestname\romannumeral\@gls@prevlevel
7278      \endcsname\space}}}{%

```

Subtract this length from the previous level's paragraph indent and set to `\glstreeindent`.

```

7279      \addtolength\parindent{-\glstreeindent}%
7280      \setlength\glstreeindent\parindent
7281      \fi
7282      \fi

```

Set the hanging indentation.

```

7283      \hangindent\glstreeindent

```

Put the name to the left of the paragraph block

```

7284      \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
7285      \textbf{\glstarget{##2}{\glossentryname{##2}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

7286      \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%

```

Do the description followed by the description terminator and location list.

```

7287      \glossentrydesc{##2}\glspostdescription\space ##3\par

```

Set the previous level macro to the current level.

```

7288      \def\@gls@prevlevel{##1}%
7289      }%

```

Vertical gap between groups is the same as that used by indices:

```

7290      \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7291      }

```

almtreegroup Like the almtree style but the glossary groups have headings.

```

7292 \newglossarystyle{almtreegroup}{%

```

Base it on the `glostylealmtree` style:

```

7293      \setglossarystyle{almtree}%

```

Give each group a heading.

```

7294      \renewcommand{\glsgroupheading}[1]{\par
7295      \def\@gls@prevlevel{-1}%
7296      \hangindent0pt\relax
7297      \parindent0pt\relax
7298      \textbf{\glsgrouptitle{##1}}\par\indexspace}%
7299      }

```

`almtreehypergroup` The `almtreehypergroup` style is like the `almtreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7300 \newglossarystyle{almtreehypergroup}{%
```

Base it on the `glostylealmtree` style:

```
7301 \setglossarystyle{almtree}%
```

Put the navigation links in the header

```
7302 \renewcommand*{\glossaryheader}{%
```

```
7303 \par
```

```
7304 \def\@gls@prevlevel{-1}%
```

```
7305 \hangindent0pt\relax
```

```
7306 \parindent0pt\relax
```

```
7307 \textbf{\glsnavigation}\par\indexspace}%
```

Put a `hypertarget` at the start of each group

```
7308 \renewcommand*{\glsgroupheading}[1]{%
```

```
7309 \par
```

```
7310 \def\@gls@prevlevel{-1}%
```

```
7311 \hangindent0pt\relax
```

```
7312 \parindent0pt\relax
```

```
7313 \textbf{\glsnavhypertarget{##1}\glsgetgrouptitle{##1}}\par
```

```
7314 \indexspace}}
```

5 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.

```
7315 \NeedsTeXFormat{LaTeX2e}
```

```
7316 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]
```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```
7317 \ifglsxindy
```

```
7318 \renewcommand*\GlsAddXdyAttribute[1]{%
```

```
7319 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
```

```
7320 \expandafter\toks@\expandafter{\@xdylocref}%
```

```
7321 \edef\@xdylocref{\the\toks@ ^^J%
```

```
7322 (markup-locref
```

```
7323 :open \string"\string~n\string\setentrycounter
```

```
7324 {\noexpand\glscounter}}%
```

```
7325 \expandafter\string\csname#1\endcsname
```

```
7326 \expandafter\@gobble\string\{\string" ^^J
```

```
7327 :close \string"\expandafter\@gobble\string\}\string" ^^J
```

```
7328 :attr \string"#1\string"))}}
```

Only has an effect before `\writeist`:

```
7329 \fi
```

\GlsAddXdyCounters

```
7330 \renewcommand*\GlsAddXdyCounters[1]{%
7331   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
7332     in compatibility mode.}%
7333 }
```

Add predefined attributes

```
7334 \GlsAddXdyAttribute{glsnumberformat}
7335 \GlsAddXdyAttribute{textrm}
7336 \GlsAddXdyAttribute{textsf}
7337 \GlsAddXdyAttribute{texttt}
7338 \GlsAddXdyAttribute{textbf}
7339 \GlsAddXdyAttribute{textmd}
7340 \GlsAddXdyAttribute{textit}
7341 \GlsAddXdyAttribute{textup}
7342 \GlsAddXdyAttribute{textsl}
7343 \GlsAddXdyAttribute{textsc}
7344 \GlsAddXdyAttribute{emph}
7345 \GlsAddXdyAttribute{glshypernumber}
7346 \GlsAddXdyAttribute{hyperrrm}
7347 \GlsAddXdyAttribute{hypersf}
7348 \GlsAddXdyAttribute{hypertt}
7349 \GlsAddXdyAttribute{hyperbf}
7350 \GlsAddXdyAttribute{hypermd}
7351 \GlsAddXdyAttribute{hyperit}
7352 \GlsAddXdyAttribute{hyperup}
7353 \GlsAddXdyAttribute{hypersl}
7354 \GlsAddXdyAttribute{hypersc}
7355 \GlsAddXdyAttribute{hyperemph}
```

\GlsAddXdyLocation Restore v2.07 definition:

```
7356 \ifglxindy
7357   \renewcommand*\GlsAddXdyLocation[2]{%
7358     \edef\@xdyuserlocationdefs{%
7359       \@xdyuserlocationdefs ^^J%
7360       (define-location-class \string"#1\string"^^J\space\space
7361         \space(#2))
7362     }%
7363     \edef\@xdyuserlocationnames{%
7364       \@xdyuserlocationnames^^J\space\space\space
7365       \string"#1\string"%
7366     }
7367 \fi
```

\@do@wrglossary

```
7368 \renewcommand*\@do@wrglossary[1]{%
  Determine whether to use xindy or makeindex syntax
7369 \ifglxindy
```

Need to determine if the formatting information starts with a (or) indicating a range.

```

7370 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
7371 \def\@glo@range{}%
7372 \expandafter\if\@glo@prefix(\relax
7373 \def\@glo@range{:open-range}%
7374 \else
7375 \expandafter\if\@glo@prefix)\relax
7376 \def\@glo@range{:close-range}%
7377 \fi
7378 \fi

```

Get the location and escape any special characters

```

7379 \protected@edef\@glslocref{\theglsentrycounter}%
7380 \@gls@checkmkidxchars\@glslocref

```

Write to the glossary file using xindy syntax.

```

7381 \glossary[\csname glo@#1@type\endcsname]{%
7382 (indexentry :tkey (\csname glo@#1@index\endcsname)
7383 :locref \string\@glslocref\string" %
7384 :attr \string\@glo@suffix\string" \@glo@range
7385 )
7386 }%
7387 \else

```

Convert the format information into the format required for makeindex

```

7388 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat

```

Write to the glossary file using makeindex syntax.

```

7389 \glossary[\csname glo@#1@type\endcsname]{%
7390 \string\glossaryentry{\csname glo@#1@index\endcsname
7391 \@gls@encapchar\@glo@numfmt}\theglsentrycounter}}%
7392 \fi
7393 }

```

\@set@glo@numformat Only had 3 arguments in v2.07

```

7394 \def\@set@glo@numformat#1#2#3{%
7395 \expandafter\@glo@check@mkidxrangechar#3\@nil
7396 \protected@edef#1{%
7397 \@glo@prefix setentrycounter[] {#2}%
7398 \expandafter\string\csname\@glo@suffix\endcsname
7399 }%
7400 \@gls@checkmkidxchars#1%
7401 }

```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```

7402 \ifglsxindy
7403 \def\writeist{%
7404 \openout\glswrite=\istfilename

```

```

7405 \write\glswrite{;; xindy style file created by the glossaries
7406   package in compatible-2.07 mode}%
7407 \write\glswrite{;; for document '\jobname' on
7408   \the\year-\the\month-\the\day}%
7409 \write\glswrite{^^J; required styles^^J}
7410 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
7411   \ifx\@xdystyle\@empty
7412   \else
7413     \protected@write\glswrite{{(require
7414       \string"\@xdystyle.xdy\string")}}%
7415   \fi
7416 }%
7417 \write\glswrite{^^J%
7418   ; list of allowed attributes (number formats)^^J}%
7419 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
7420 \write\glswrite{^^J; user defined alphabets^^J}%
7421 \write\glswrite{\@xdyuseralphabets}%
7422 \write\glswrite{^^J; location class definitions^^J}%
7423 \protected@edef\@gls@roman{\@roman{0}\string"
7424   \string"roman-numbers-lowercase\string" :sep \string"}}%
7425 \@onelevel@sanitize\@gls@roman
7426 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
7427   :sep \string"}}%
7428 \@onelevel@sanitize\@tmp
7429 \ifx\@tmp\@gls@roman
7430   \write\glswrite{(define-location-class
7431     \string"roman-page-numbers\string"^^J\space\space\space
7432     (\string"roman-numbers-lowercase\string")
7433     :min-range-length \@glsminrange)}}%
7434 \else
7435   \write\glswrite{(define-location-class
7436     \string"roman-page-numbers\string"^^J\space\space\space
7437     (:sep "\@gls@roman")
7438     :min-range-length \@glsminrange)}}%
7439 \fi
7440 \write\glswrite{(define-location-class
7441   \string"Roman-page-numbers\string"^^J\space\space\space
7442   (\string"roman-numbers-uppercase\string")
7443   :min-range-length \@glsminrange)}}%
7444 \write\glswrite{(define-location-class
7445   \string"arabic-page-numbers\string"^^J\space\space\space
7446   (\string"arabic-numbers\string")
7447   :min-range-length \@glsminrange)}}%
7448 \write\glswrite{(define-location-class
7449   \string"alpha-page-numbers\string"^^J\space\space\space
7450   (\string"alpha\string")
7451   :min-range-length \@glsminrange)}}%
7452 \write\glswrite{(define-location-class
7453   \string"Alpha-page-numbers\string"^^J\space\space\space

```

```

7454      (\string"ALPHA\string")
7455      :min-range-length \@glsmminrange))}%
7456 \write\glswrite{(define-location-class
7457   \string"Appendix-page-numbers\string"^^J\space\space\space
7458   (\string"ALPHA\string"
7459    :sep \string"\@glAlphacompositor\string"
7460    \string"arabic-numbers\string")
7461    :min-range-length \@glsmminrange))}%
7462 \write\glswrite{(define-location-class
7463   \string"arabic-section-numbers\string"^^J\space\space\space
7464   (\string"arabic-numbers\string"
7465    :sep \string"\glscocompositor\string"
7466    \string"arabic-numbers\string")
7467    :min-range-length \@glsmminrange))}%
7468 \write\glswrite{^^J; user defined location classes}%
7469 \write\glswrite{@xdyuserlocationdefs}%
7470 \write\glswrite{^^J; define cross-reference class^^J}%
7471 \write\glswrite{(define-crossref-class \string"see\string"
7472   :unverified )}%
7473 \write\glswrite{(markup-crossref-list
7474   :class \string"see\string"^^J\space\space\space
7475   :open \string"\string\glseeformat\string"
7476   :close \string"{}\string")}%
7477 \write\glswrite{^^J; define the order of the location classes}%
7478 \write\glswrite{(define-location-class-order
7479   (\@xdylocationclassorder))}%
7480 \write\glswrite{^^J; define the glossary markup^^J}%
7481 \write\glswrite{(markup-index^^J\space\space\space
7482   :open \string"\string
7483     \glossarysection[\string\glossarytoctitle]{\string
7484     \glossarytitle}\string\glossarypreamble\string~n\string\begin
7485     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
7486     \space\space:close \string"\expandafter\@gobble
7487     \string\%\string~n\string
7488     \end{theglossary}\string\glossarypostamble
7489     \string~n\string" ^^J\space\space\space
7490     :tree)}}%
7491 \write\glswrite{(markup-letter-group-list
7492   :sep \string"\string\glsgroupskip\string~n\string")}%
7493 \write\glswrite{(markup-indexentry
7494   :open \string"\string\relax \string\glresetentrylist
7495     \string~n\string")}%
7496 \write\glswrite{(markup-locclass-list :open
7497   \string"\glsoopenbrace\string\glossaryentrynumbers
7498   \glsoopenbrace\string\relax\space \string"^^J\space\space\space
7499   :sep \string", \string"
7500   :close \string"\glsclosebrace\glsclosebrace\string")}%
7501 \write\glswrite{(markup-locref-list
7502   :sep \string"\string\delimN\space\string")}%

```



```

7503 \write\glswrite{(markup-range
7504 :sep \string"\string\delimR\space\string")}%
7505 \@onelevel@sanitize\gls@suffixF
7506 \@onelevel@sanitize\gls@suffixFF
7507 \ifx\gls@suffixF\@empty
7508 \else
7509 \write\glswrite{(markup-range
7510 :close "\gls@suffixF" :length 1 :ignore-end)}%
7511 \fi
7512 \ifx\gls@suffixFF\@empty
7513 \else
7514 \write\glswrite{(markup-range
7515 :close "\gls@suffixFF" :length 2 :ignore-end)}%
7516 \fi
7517 \write\glswrite{^^J; define format to use for locations^^J}%
7518 \write\glswrite{\@xdylocref}%
7519 \write\glswrite{^^J; define letter group list format^^J}%
7520 \write\glswrite{(markup-letter-group-list
7521 :sep \string"\string\glsgroupskip\string~\n\string")}%
7522 \write\glswrite{^^J; letter group headings^^J}%
7523 \write\glswrite{(markup-letter-group
7524 :open-head \string"\string\glsgroupheading
7525 \glsopenbrace\string"^^J\space\space\space
7526 :close-head \string"\glsclosebrace\string")}%
7527 \write\glswrite{^^J; additional letter groups^^J}%
7528 \write\glswrite{\@xdylettergroups}%
7529 \write\glswrite{^^J; additional sort rules^^J}%
7530 \write\glswrite{\@xdysortrules}%
7531 \noist}
7532 \else
7533 \edef\@gls@actualchar{\string?}
7534 \edef\@gls@encapchar{\string|}
7535 \edef\@gls@levelchar{\string!}
7536 \edef\@gls@quotechar{\string"}
7537 \def\writeist{\relax
7538 \openout\glswrite=\istfilename
7539 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
7540 created by the glossaries package}
7541 \write\glswrite{\expandafter\@gobble\string\% for document
7542 '\jobname' on \the\year-\the\month-\the\day}
7543 \write\glswrite{actual '\@gls@actualchar'}
7544 \write\glswrite{encap '\@gls@encapchar'}
7545 \write\glswrite{level '\@gls@levelchar'}
7546 \write\glswrite{quote '\@gls@quotechar'}
7547 \write\glswrite{keyword \string"\string\glossaryentry\string"}
7548 \write\glswrite{preamble \string"\string\glossarysection[\string
7549 \glossarytoctitle]{\string\glossarytitle}\string
7550 \glossarypreamble\string\n\string\begin{theglossary}\string
7551 \glossaryheader\string\n\string"}

```

```

7552 \write\glswrite{postamble \string"\string%\string\n\string
7553 \end{theglossary}\string\glossarypostamble\string\n
7554 \string"}
7555 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
7556 \string"}
7557 \write\glswrite{item_0 \string"\string%\string\n\string"}
7558 \write\glswrite{item_1 \string"\string%\string\n\string"}
7559 \write\glswrite{item_2 \string"\string%\string\n\string"}
7560 \write\glswrite{item_01 \string"\string%\string\n\string"}
7561 \write\glswrite{item_x1
7562 \string"\string\relax \string\glresetentrylist\string\n
7563 \string"}
7564 \write\glswrite{item_12 \string"\string%\string\n\string"}
7565 \write\glswrite{item_x2
7566 \string"\string\relax \string\glresetentrylist\string\n
7567 \string"}
7568 \write\glswrite{delim_0 \string"\string\{\string
7569 \glossaryentrynumbers\string\{\string\relax \string"}
7570 \write\glswrite{delim_1 \string"\string\{\string
7571 \glossaryentrynumbers\string\{\string\relax \string"}
7572 \write\glswrite{delim_2 \string"\string\{\string
7573 \glossaryentrynumbers\string\{\string\relax \string"}
7574 \write\glswrite{delim_t \string"\string\}\string\}\string"}
7575 \write\glswrite{delim_n \string"\string\delimN \string"}
7576 \write\glswrite{delim_r \string"\string\delimR \string"}
7577 \write\glswrite{headings_flag 1}
7578 \write\glswrite{heading_prefix
7579 \string"\string\glsgroupheading\string\{\string"}
7580 \write\glswrite{heading_suffix
7581 \string"\string\}\string\relax
7582 \string\glresetentrylist \string"}
7583 \write\glswrite{symhead_positive \string"glssymbols\string"}
7584 \write\glswrite{numhead_positive \string"glsnnumbers\string"}
7585 \write\glswrite{page_compositor \string"glscpositor\string"}
7586 \@glscbsdq\glscsuffixF
7587 \@glscbsdq\glscsuffixFF
7588 \ifx\glscsuffixF\empty
7589 \else
7590 \write\glswrite{suffix_2p \string"\glscsuffixF\string"}
7591 \fi
7592 \ifx\glscsuffixFF\empty
7593 \else
7594 \write\glswrite{suffix_3p \string"\glscsuffixFF\string"}
7595 \fi
7596 \noist
7597 }
7598 \fi

```

\noist

```
7599 \renewcommand*{\noist}{\let\writeist\relax}
```

Compatibility macros.

```
7600 \NeedsTeXFormat{LaTeX2e}
```

```
7601 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]
```

Compatibility macros for predefined glossary styles:

`compatglossarystyle` Defines a compatibility glossary style.

```
7602 \newcommand{\compatglossarystyle}[2]{%
7603   \ifcsundef{@glscompstyle@#1}%
7604   {%
7605     \csdef{@glscompstyle@#1}{#2}%
7606   }%
7607   {%
7608     \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
7609   }%
7610 }
```

Backward compatible inline style.

```
7611 \compatglossarystyle{inline}{%
7612   \renewcommand{\glossaryentryfield}[5]{%
7613     \glsinlinedopostchild
7614     \gls@inlinesep
7615     \def\glo@desc{##3}%
7616     \def\@no@post@desc{\nopostdesc}%
7617     \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
7618     \ifx\glo@desc\@no@post@desc
7619       \glsinlineemptydescformat{##4}{##5}%
7620     \else
7621       \ifstrepty{##3}%
7622         {\glsinlineemptydescformat{##4}{##5}}%
7623         {\glsinlinedescformat{##3}{##4}{##5}}%
7624       \fi
7625       \ifglshaschildren{##1}%
7626       {%
7627         \glsresetsubentrycounter
7628         \glsinlineparentchildseparator
7629         \def\gls@inlinesubsep{}%
7630         \def\gls@inlinepostchild{\glsinlinepostchild}%
7631       }%
7632     }%
7633     \def\gls@inlinesep{\glsinlineseparator}%
7634   }%
```

Sub-entries display description:

```
7635 \renewcommand{\glossarysubentryfield}[6]{%
7636   \gls@inlinesubsep%
7637   \glsinlinesubnameformat{##2}{##3}%
7638   \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
```

```

7639 \def\gls@inlinesubsep{\glsinlinesubseparator}%
7640 }%
7641 }

```

Backward compatible list style.

```

7642 \compatglossarystyle{list}{%
7643 \renewcommand*{\glossaryentryfield}[5]{%
7644 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
7645 ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```

7646 \renewcommand*{\glossarysubentryfield}[6]{%
7647 \glssubentryitem{##2}%
7648 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
7649 }

```

Backward compatible listgroup style.

```

7650 \compatglossarystyle{listgroup}{%
7651 \csuse{@glscompstyle@list}%
7652 }%

```

Backward compatible listhypergroup style.

```

7653 \compatglossarystyle{listhypergroup}{%
7654 \csuse{@glscompstyle@list}%
7655 }%

```

Backward compatible altlist style.

```

7656 \compatglossarystyle{altlist}{%
7657 \renewcommand*{\glossaryentryfield}[5]{%
7658 \item[\glsentryitem{##1}\glstarget{##1}{##2}]%
7659 \mbox{\par\nobreak\@afterheading
7660 ##3\glspostdescription\space ##5}%
7661 \renewcommand{\glossarysubentryfield}[6]{%
7662 \par
7663 \glssubentryitem{##2}%
7664 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
7665 }%

```

Backward compatible altlistgroup style.

```

7666 \compatglossarystyle{altlistgroup}{%
7667 \csuse{@glscompstyle@altlist}%
7668 }%

```

Backward compatible altlisthypergroup style.

```

7669 \compatglossarystyle{altlisthypergroup}{%
7670 \csuse{@glscompstyle@altlist}%
7671 }%

```

Backward compatible listdotted style.

```

7672 \compatglossarystyle{listdotted}{%
7673 \renewcommand*{\glossaryentryfield}[5]{%
7674 \item[\makebox[\glslistdottedwidth][l]{%

```

```

7675      \glstryitem{##1}\glstarget{##1}{##2}%
7676      \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
7677 \renewcommand*{\glossarysubentryfield}[6]{%
7678   \item[]\makebox[\glslstdottedwidth][l]{%
7679     \glssubentryitem{##2}%
7680     \glstarget{##2}{##3}%
7681     \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
7682 }%

```

Backward compatible sublistdotted style.

```

7683 \compatglossarystyle{sublistdotted}{%
7684   \csuse{@glscmpstyle@listdotted}%
7685   \renewcommand*{\glossaryentryfield}[5]{%
7686     \item[\glstryitem{##1}\glstarget{##1}{##2}]}%
7687 }%

```

Backward compatible long style.

```

7688 \compatglossarystyle{long}{%
7689   \renewcommand*{\glossaryentryfield}[5]{%
7690     \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
7691   \renewcommand*{\glossarysubentryfield}[6]{%
7692     &
7693     \glssubentryitem{##2}%
7694     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
7695 }%

```

Backward compatible longborder style.

```

7696 \compatglossarystyle{longborder}{%
7697   \csuse{@glscmpstyle@long}%
7698 }%

```

Backward compatible longheader style.

```

7699 \compatglossarystyle{longheader}{%
7700   \csuse{@glscmpstyle@long}%
7701 }%

```

Backward compatible longheaderborder style.

```

7702 \compatglossarystyle{longheaderborder}{%
7703   \csuse{@glscmpstyle@long}%
7704 }%

```

Backward compatible long3col style.

```

7705 \compatglossarystyle{long3col}{%
7706   \renewcommand*{\glossaryentryfield}[5]{%
7707     \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
7708   \renewcommand*{\glossarysubentryfield}[6]{%
7709     &
7710     \glssubentryitem{##2}%
7711     \glstarget{##2}{\strut}##4 & ##6\\}%
7712 }%

```

Backward compatible long3colborder style.

```

7713 \compatglossarystyle{long3colborder}{%
7714 \csuse{@glscompstyle@long3col}%
7715 }%

```

Backward compatible long3colheader style.

```

7716 \compatglossarystyle{long3colheader}{%
7717 \csuse{@glscompstyle@long3col}%
7718 }%

```

Backward compatible long3colheaderborder style.

```

7719 \compatglossarystyle{long3colheaderborder}{%
7720 \csuse{@glscompstyle@long3col}%
7721 }%

```

Backward compatible long4col style.

```

7722 \compatglossarystyle{long4col}{%
7723 \renewcommand*{\glossaryentryfield}[5]{%
7724 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
7725 \renewcommand*{\glossarysubentryfield}[6]{%
7726 &
7727 \glssubentryitem{##2}%
7728 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
7729 }%

```

Backward compatible long4colheader style.

```

7730 \compatglossarystyle{long4colheader}{%
7731 \csuse{@glscompstyle@long4col}%
7732 }%

```

Backward compatible long4colborder style.

```

7733 \compatglossarystyle{long4colborder}{%
7734 \csuse{@glscompstyle@long4col}%
7735 }%

```

Backward compatible long4colheaderborder style.

```

7736 \compatglossarystyle{long4colheaderborder}{%
7737 \csuse{@glscompstyle@long4col}%
7738 }%

```

Backward compatible altlong4col style.

```

7739 \compatglossarystyle{altlong4col}{%
7740 \csuse{@glscompstyle@long4col}%
7741 }%

```

Backward compatible altlong4colheader style.

```

7742 \compatglossarystyle{altlong4colheader}{%
7743 \csuse{@glscompstyle@long4col}%
7744 }%

```

Backward compatible altlong4colborder style.

```

7745 \compatglossarystyle{altlong4colborder}{%
7746 \csuse{@glscompstyle@long4col}%
7747 }%

```

Backward compatible altlong4colheaderborder style.

```
7748 \compatglossarystyle{altlong4colheaderborder}{%
7749 \csuse{@glscmpstyle@long4col}%
7750 }%
```

Backward compatible long style.

```
7751 \compatglossarystyle{longragged}{%
7752 \renewcommand*{\glossaryentryfield}[5]{%
7753 \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
7754 \tabularnewline}%
7755 \renewcommand*{\glossarysubentryfield}[6]{%
7756 &
7757 \glssubentryitem{##2}%
7758 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
7759 \tabularnewline}%
7760 }%
```

Backward compatible longraggedborder style.

```
7761 \compatglossarystyle{longraggedborder}{%
7762 \csuse{@glscmpstyle@longragged}%
7763 }%
```

Backward compatible longraggedheader style.

```
7764 \compatglossarystyle{longraggedheader}{%
7765 \csuse{@glscmpstyle@longragged}%
7766 }%
```

Backward compatible longraggedheaderborder style.

```
7767 \compatglossarystyle{longraggedheaderborder}{%
7768 \csuse{@glscmpstyle@longragged}%
7769 }%
```

Backward compatible longragged3col style.

```
7770 \compatglossarystyle{longragged3col}{%
7771 \renewcommand*{\glossaryentryfield}[5]{%
7772 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
7773 \renewcommand*{\glossarysubentryfield}[6]{%
7774 &
7775 \glssubentryitem{##2}%
7776 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
7777 }%
```

Backward compatible longragged3colborder style.

```
7778 \compatglossarystyle{longragged3colborder}{%
7779 \csuse{@glscmpstyle@longragged3col}%
7780 }%
```

Backward compatible longragged3colheader style.

```
7781 \compatglossarystyle{longragged3colheader}{%
7782 \csuse{@glscmpstyle@longragged3col}%
7783 }%
```

Backward compatible longragged3colheaderborder style.

```
7784 \compatglossarystyle{longragged3colheaderborder}{%
7785 \csuse{@glscmpstyle@longragged3col}%
7786 }%
```

Backward compatible altlongragged4col style.

```
7787 \compatglossarystyle{altlongragged4col}{%
7788 \renewcommand*{\glossaryentryfield}[5]{%
7789 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
7790 \renewcommand*{\glossarysubentryfield}[6]{%
7791 &
7792 \glssubentryitem{##2}%
7793 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
7794 }%
```

Backward compatible altlongragged4colheader style.

```
7795 \compatglossarystyle{altlongragged4colheader}{%
7796 \csuse{@glscmpstyle@altlong4col}%
7797 }%
```

Backward compatible altlongragged4colborder style.

```
7798 \compatglossarystyle{altlongragged4colborder}{%
7799 \csuse{@glscmpstyle@altlong4col}%
7800 }%
```

Backward compatible altlongragged4colheaderborder style.

```
7801 \compatglossarystyle{altlongragged4colheaderborder}{%
7802 \csuse{@glscmpstyle@altlong4col}%
7803 }%
```

Backward compatible index style.

```
7804 \compatglossarystyle{index}{%
7805 \renewcommand*{\glossaryentryfield}[5]{%
7806 \item\glstryitem{##1}\textbf{\glstarget{##1}{##2}}%
7807 \ifx\relax##4\relax
7808 \else
7809 \space{##4}%
7810 \fi
7811 \space ##3\glspostdescription \space ##5}%
7812 \renewcommand*{\glossarysubentryfield}[6]{%
7813 \ifcase##1\relax
7814 % level 0
7815 \item
7816 \or
7817 % level 1
7818 \subitem
7819 \glssubentryitem{##2}%
7820 \else
7821 % all other levels
7822 \subsubitem
7823 \fi
```



```

7824 \textbf{\glstarget{##2}{##3}}%
7825 \ifx\relax##5\relax
7826 \else
7827 \space(##5)%
7828 \fi
7829 \space##4\glspostdescription\space ##6}%
7830 }%

```

Backward compatible indexgroup style.

```

7831 \compatglossarystyle{indexgroup}{%
7832 \csuse{@glscmpstyle@index}%
7833 }%

```

Backward compatible indexhypergroup style.

```

7834 \compatglossarystyle{indexhypergroup}{%
7835 \csuse{@glscmpstyle@index}%
7836 }%

```

Backward compatible tree style.

```

7837 \compatglossarystyle{tree}{%
7838 \renewcommand{\glossaryentryfield}[5]{%
7839 \hangindent0pt\relax
7840 \parindent0pt\relax
7841 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
7842 \ifx\relax##4\relax
7843 \else
7844 \space(##4)%
7845 \fi
7846 \space ##3\glspostdescription \space ##5\par}%
7847 \renewcommand{\glossarysubentryfield}[6]{%
7848 \hangindent##1\glstreeindent\relax
7849 \parindent##1\glstreeindent\relax
7850 \ifnum##1=1\relax
7851 \glssubentryitem{##2}%
7852 \fi
7853 \textbf{\glstarget{##2}{##3}}%
7854 \ifx\relax##5\relax
7855 \else
7856 \space(##5)%
7857 \fi
7858 \space##4\glspostdescription\space ##6\par}%
7859 }%

```

Backward compatible treegroup style.

```

7860 \compatglossarystyle{treegroup}{%
7861 \csuse{@glscmpstyle@tree}%
7862 }%

```

Backward compatible treehypergroup style.

```

7863 \compatglossarystyle{treehypergroup}{%
7864 \csuse{@glscmpstyle@tree}%
7865 }%

```

Backward compatible treenoname style.

```

7866 \compatglossarystyle{treenoname}{%
7867   \renewcommand{\glossaryentryfield}[5]{%
7868     \hangindent0pt\relax
7869     \parindent0pt\relax
7870     \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}}%
7871     \ifx\relax##4\relax
7872     \else
7873       \space{##4}%
7874     \fi
7875     \space ##3\glspostdescription \space ##5\par}%
7876 \renewcommand{\glossarysubentryfield}[6]{%
7877   \hangindent##1\glstreeindent\relax
7878   \parindent##1\glstreeindent\relax
7879   \ifnum##1=1\relax
7880     \glssubentryitem{##2}%
7881   \fi
7882   \glstarget{##2}{\strut}%
7883   ##4\glspostdescription\space ##6\par}%
7884 }%

```

Backward compatible treenonamegroup style.

```

7885 \compatglossarystyle{treenonamegroup}{%
7886   \csuse{@glscmpstyle@treenoname}%
7887 }%

```

Backward compatible treenonamehypergroup style.

```

7888 \compatglossarystyle{treenonamehypergroup}{%
7889   \csuse{@glscmpstyle@treenoname}%
7890 }%

```

Backward compatible alttree style.

```

7891 \compatglossarystyle{alttree}{%
7892   \renewcommand{\glossaryentryfield}[5]{%
7893     \ifnum\@gls@prevlevel=0\relax
7894     \else
7895       \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
7896       \hangindent\glstreeindent
7897       \parindent\glstreeindent
7898     \fi
7899     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
7900       \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}}%
7901     \ifx\relax##4\relax
7902     \else
7903       (##4)\space
7904     \fi
7905     ##3\glspostdescription \space ##5\par
7906     \def\@gls@prevlevel{0}%
7907   }%
7908   \renewcommand{\glossarysubentryfield}[6]{%

```

```

7909 \ifnum##1=1\relax
7910 \glssubentryitem{##2}%
7911 \fi
7912 \ifnum\@gls@prevlevel=##1\relax
7913 \else
7914 \ifundefined{\@glswidestname\romannumeral##1}{%
7915 \settowidth{\@gls@tmplen}{\textbf{\@glswidestname\space}}{%
7916 \settowidth{\@gls@tmplen}{\textbf{%
7917 \csname \@glswidestname\romannumeral##1\endcsname\space}}}%
7918 \ifnum\@gls@prevlevel<##1\relax
7919 \setlength\glstreeindent\@gls@tmplen
7920 \addtolength\glstreeindent\parindent
7921 \parindent\glstreeindent
7922 \else
7923 \ifundefined{\@glswidestname\romannumeral\@gls@prevlevel}{%
7924 \settowidth{\glstreeindent}{\textbf{%
7925 \@glswidestname\space}}{%
7926 \settowidth{\glstreeindent}{\textbf{%
7927 \csname \@glswidestname\romannumeral\@gls@prevlevel
7928 \endcsname\space}}}%
7929 \addtolength\parindent{-\glstreeindent}%
7930 \setlength\glstreeindent\parindent
7931 \fi
7932 \fi
7933 \hangindent\glstreeindent
7934 \makebox[0pt][r]{\makebox[\@gls@tmplen][l]{%
7935 \textbf{\glstarget{##2}{##3}}}%
7936 \ifx##5\relax\relax
7937 \else
7938 (##5)\space
7939 \fi
7940 ##4\glspostdescription\space ##6\par
7941 \def\@gls@prevlevel{##1}%
7942 }%
7943 }%

```

Backward compatible alttreegroup style.

```

7944 \compatglossarystyle{alttreegroup}{%
7945 \csuse{\@glscompstyle@almtree}%
7946 }%

```

Backward compatible alttreehypergroup style.

```

7947 \compatglossarystyle{alttreehypergroup}{%
7948 \csuse{\@glscompstyle@almtree}%
7949 }%

```

Backward compatible mcolindex style.

```

7950 \compatglossarystyle{mcolindex}{%
7951 \csuse{\@glscompstyle@index}%
7952 }%

```

Backward compatible mcolindexgroup style.

```
7953 \compatglossarystyle{mcolindexgroup}{%  
7954 \csuse{@glscompstyle@index}%  
7955 }%
```

Backward compatible mcolindexhypergroup style.

```
7956 \compatglossarystyle{mcolindexhypergroup}{%  
7957 \csuse{@glscompstyle@index}%  
7958 }%
```

Backward compatible mcoltree style.

```
7959 \compatglossarystyle{mcoltree}{%  
7960 \csuse{@glscompstyle@tree}%  
7961 }%
```

Backward compatible mcoltreegroup style.

```
7962 \compatglossarystyle{mcolindextreegroup}{%  
7963 \csuse{@glscompstyle@tree}%  
7964 }%
```

Backward compatible mcoltreehypergroup style.

```
7965 \compatglossarystyle{mcolindextreehypergroup}{%  
7966 \csuse{@glscompstyle@tree}%  
7967 }%
```

Backward compatible mcoltreenoname style.

```
7968 \compatglossarystyle{mcoltreenoname}{%  
7969 \csuse{@glscompstyle@tree}%  
7970 }%
```

Backward compatible mcoltreenonamegroup style.

```
7971 \compatglossarystyle{mcoltreenonamegroup}{%  
7972 \csuse{@glscompstyle@tree}%  
7973 }%
```

Backward compatible mcoltreenonamehypergroup style.

```
7974 \compatglossarystyle{mcoltreenonamehypergroup}{%  
7975 \csuse{@glscompstyle@tree}%  
7976 }%
```

Backward compatible mcolalttree style.

```
7977 \compatglossarystyle{mcolalttree}{%  
7978 \csuse{@glscompstyle@alttree}%  
7979 }%
```

Backward compatible mcolalttreegroup style.

```
7980 \compatglossarystyle{mcolalttreegroup}{%  
7981 \csuse{@glscompstyle@alttree}%  
7982 }%
```

Backward compatible mcolalttreehypergroup style.

```
7983 \compatglossarystyle{mcolalttreehypergroup}{%  
7984 \csuse{@glscompstyle@alttree}%  
7985 }%
```

Backward compatible superragged style.

```
7986 \compatglossarystyle{superragged}{%
7987   \renewcommand*{\glossaryentryfield}[5]{%
7988     \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
7989     \tabularnewline}%
7990   \renewcommand*{\glossarysubentryfield}[6]{%
7991     &
7992     \glssubentryitem{##2}%
7993     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
7994     \tabularnewline}%
7995 }%
```

Backward compatible superraggedborder style.

```
7996 \compatglossarystyle{superraggedborder}{%
7997   \csuse{@glscmpstyle@superragged}%
7998 }%
```

Backward compatible superraggedheader style.

```
7999 \compatglossarystyle{superraggedheader}{%
8000   \csuse{@glscmpstyle@superragged}%
8001 }%
```

Backward compatible superraggedheaderborder style.

```
8002 \compatglossarystyle{superraggedheaderborder}{%
8003   \csuse{@glscmpstyle@superragged}%
8004 }%
```

Backward compatible superragged3col style.

```
8005 \compatglossarystyle{superragged3col}{%
8006   \renewcommand*{\glossaryentryfield}[5]{%
8007     \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
8008   \renewcommand*{\glossarysubentryfield}[6]{%
8009     &
8010     \glssubentryitem{##2}%
8011     \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8012 }%
```

Backward compatible superragged3colborder style.

```
8013 \compatglossarystyle{superragged3colborder}{%
8014   \csuse{@glscmpstyle@superragged3col}%
8015 }%
```

Backward compatible superragged3colheader style.

```
8016 \compatglossarystyle{superragged3colheader}{%
8017   \csuse{@glscmpstyle@superragged3col}%
8018 }%
```

Backward compatible superragged3colheaderborder style.

```
8019 \compatglossarystyle{superragged3colheaderborder}{%
8020   \csuse{@glscmpstyle@superragged3col}%
8021 }%
```

Backward compatible altsuperragged4col style.

```
8022 \compatglossarystyle{altsuperragged4col}{%
8023   \renewcommand*{\glossaryentryfield}[5]{%
8024     \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8025   \renewcommand*{\glossarysubentryfield}[6]{%
8026     &
8027     \glssubentryitem{##2}%
8028     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8029 }%
```

Backward compatible altsuperragged4colheader style.

```
8030 \compatglossarystyle{altsuperragged4colheader}{%
8031   \csuse{@glscompstyle@altsuperragged4col}%
8032 }%
```

Backward compatible altsuperragged4colborder style.

```
8033 \compatglossarystyle{altsuperragged4colborder}{%
8034   \csuse{@glscompstyle@altsuperragged4col}%
8035 }%
```

Backward compatible altsuperragged4colheaderborder style.

```
8036 \compatglossarystyle{altsuperragged4colheaderborder}{%
8037   \csuse{@glscompstyle@altsuperragged4col}%
8038 }%
```

Backward compatible super style.

```
8039 \compatglossarystyle{super}{%
8040   \renewcommand*{\glossaryentryfield}[5]{%
8041     \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8042   \renewcommand*{\glossarysubentryfield}[6]{%
8043     &
8044     \glssubentryitem{##2}%
8045     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8046 }%
```

Backward compatible superborder style.

```
8047 \compatglossarystyle{superborder}{%
8048   \csuse{@glscompstyle@super}%
8049 }%
```

Backward compatible superheader style.

```
8050 \compatglossarystyle{superheader}{%
8051   \csuse{@glscompstyle@super}%
8052 }%
```

Backward compatible superheaderborder style.

```
8053 \compatglossarystyle{superheaderborder}{%
8054   \csuse{@glscompstyle@super}%
8055 }%
```

Backward compatible super3col style.

```
8056 \compatglossarystyle{super3col}{%
```

```

8057 \renewcommand*{\glossaryentryfield}[5]{%
8058   \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8059 \renewcommand*{\glossarysubentryfield}[6]{%
8060   &
8061   \glssubentryitem{##2}%
8062   \glstarget{##2}{\strut}##4 & ##6\\}%
8063 }%

Backward compatible super3colborder style.
8064 \compatglossarystyle{super3colborder}{%
8065 \csuse{@glscmpstyle@super3col}%
8066 }%

Backward compatible super3colheader style.
8067 \compatglossarystyle{super3colheader}{%
8068 \csuse{@glscmpstyle@super3col}%
8069 }%

Backward compatible super3colheaderborder style.
8070 \compatglossarystyle{super3colheaderborder}{%
8071 \csuse{@glscmpstyle@super3col}%
8072 }%

Backward compatible super4col style.
8073 \compatglossarystyle{super4col}{%
8074 \renewcommand*{\glossaryentryfield}[5]{%
8075   \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8076 \renewcommand*{\glossarysubentryfield}[6]{%
8077   &
8078   \glssubentryitem{##2}%
8079   \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8080 }%

Backward compatible super4colheader style.
8081 \compatglossarystyle{super4colheader}{%
8082 \csuse{@glscmpstyle@super4col}%
8083 }%

Backward compatible super4colborder style.
8084 \compatglossarystyle{super4colborder}{%
8085 \csuse{@glscmpstyle@super4col}%
8086 }%

Backward compatible super4colheaderborder style.
8087 \compatglossarystyle{super4colheaderborder}{%
8088 \csuse{@glscmpstyle@super4col}%
8089 }%

Backward compatible altsuper4col style.
8090 \compatglossarystyle{altsuper4col}{%
8091 \csuse{@glscmpstyle@super4col}%
8092 }%

```

Backward compatible altsuper4colheader style.

```
8093 \compatglossarystyle{altsuper4colheader}{%  
8094 \csuse{@glscompstyle@super4col}%  
8095 }%
```

Backward compatible altsuper4colborder style.

```
8096 \compatglossarystyle{altsuper4colborder}{%  
8097 \csuse{@glscompstyle@super4col}%  
8098 }%
```

Backward compatible altsuper4colheaderborder style.

```
8099 \compatglossarystyle{altsuper4colheaderborder}{%  
8100 \csuse{@glscompstyle@super4col}%  
8101 }%
```

6 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.

```
8102 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```
8103 \ProvidesPackage{glossaries-accsupp}[2013/11/14 v4.0 (NLCT)]  
8104 Experimental glossaries accessibility]
```

Pass all options to glossaries:

```
8105 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
8106 \ProcessOptions
```

Override style compatibility macros:

```
8107 \newcommand*{\compatibleglossentry}[2]{%  
8108 \toks@{#2}%  
8109 \protected@edef\@do@glossentry{%  
8110 \noexpand\accsuppglossaryentryfield{#1}%  
8111 {\noexpand\glsnamefont  
8112 {\expandafter\expandonce\csname glo@#1@name\endcsname}}%  
8113 {\expandafter\expandonce\csname glo@#1@desc\endcsname}%  
8114 {\expandafter\expandonce\csname glo@#1@symbol\endcsname}%  
8115 {\the\toks@}%  
8116 }%  
8117 \@do@glossentry  
8118 }  
  
8119 \newcommand*{\compatiblesubglossentry}[3]{%  
8120 \toks@{#3}%  
8121 \protected@edef\@do@subglossentry{%
```



```

8122 \noexpand\accsuppglossarysubentryfield{\number#1}%
8123 {#2}%
8124 {\noexpand\glsnamefont
8125   {\expandafter\expandonce\csname glo@#2@name\endcsname}}}%
8126 {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
8127 {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
8128 {\the\toks@}%
8129 }%
8130 \@do@subglossentry
8131 }

```

Required packages:

```

8132 \RequirePackage{glossaries}
8133 \RequirePackage{accsupp}

```

6.1 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:

```

8134 \define@key{glossentry}{access}{%
8135   \def\@glo@access{#1}%
8136 }

```

textaccess The replacement text corresponding to the text key:

```

8137 \define@key{glossentry}{textaccess}{%
8138   \def\@glo@textaccess{#1}%
8139 }

```

firstaccess The replacement text corresponding to the first key:

```

8140 \define@key{glossentry}{firstaccess}{%
8141   \def\@glo@firstaccess{#1}%
8142 }

```

pluralaccess The replacement text corresponding to the plural key:

```

8143 \define@key{glossentry}{pluralaccess}{%
8144   \def\@glo@pluralaccess{#1}%
8145 }

```

firstpluralaccess The replacement text corresponding to the firstplural key:

```

8146 \define@key{glossentry}{firstpluralaccess}{%
8147   \def\@glo@firstpluralaccess{#1}%
8148 }

```

symbolaccess The replacement text corresponding to the symbol key:

```
8149 \define@key{glossentry}{symbolaccess}{%
8150   \def\@glo@symbolaccess{#1}%
8151 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
8152 \define@key{glossentry}{symbolpluralaccess}{%
8153   \def\@glo@symbolpluralaccess{#1}%
8154 }
```

descriptionaccess The replacement text corresponding to the description key:

```
8155 \define@key{glossentry}{descriptionaccess}{%
8156   \def\@glo@descaccess{#1}%
8157 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
8158 \define@key{glossentry}{descriptionpluralaccess}{%
8159   \def\@glo@descpluralaccess{#1}%
8160 }
```

shortaccess The replacement text corresponding to the short key:

```
8161 \define@key{glossentry}{shortaccess}{%
8162   \def\@glo@shortaccess{#1}%
8163 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
8164 \define@key{glossentry}{shortpluralaccess}{%
8165   \def\@glo@shortpluralaccess{#1}%
8166 }
```

longaccess The replacement text corresponding to the long key:

```
8167 \define@key{glossentry}{longaccess}{%
8168   \def\@glo@longaccess{#1}%
8169 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
8170 \define@key{glossentry}{longpluralaccess}{%
8171   \def\@glo@longpluralaccess{#1}%
8172 }
```

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={\glsacccsupp{inches}{in}}.

Append these new keys to \@gls@keymap:

```
8173 \appto\@gls@keymap{,%
8174   {access}{access},%
8175   {textaccess}{textaccess},%
8176   {firstaccess}{firstaccess},%
```

```

8177 {pluralaccess}{pluralaccess},%
8178 {firstpluralaccess}{firstpluralaccess},%
8179 {symbolaccess}{symbolaccess},%
8180 {symbolpluralaccess}{symbolpluralaccess},%
8181 {descaccess}{descaccess},%
8182 {descpluralaccess}{descpluralaccess},%
8183 {shortaccess}{shortaccess},%
8184 {shortpluralaccess}{shortpluralaccess},%
8185 {longaccess}{longaccess},%
8186 {longpluralaccess}{longpluralaccess}%
8187 }

```

\@gls@noaccess Indicates that no replacement text has been provided.

```

8188 \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):

```

8189 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
8190 \renewcommand*{\@newglossaryentryprehook}{%
8191   \@gls@oldnewglossaryentryprehook
8192   \def\@glo@access{\@glo@symbol}%

```

Initialise the other keys:

```

8193   \def\@glo@textaccess{\@glo@access}%
8194   \def\@glo@firstaccess{\@glo@access}%
8195   \def\@glo@pluralaccess{\@glo@textaccess}%
8196   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
8197   \def\@glo@symbolaccess{\relax}%
8198   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
8199   \def\@glo@descaccess{\relax}%
8200   \def\@glo@descpluralaccess{\@glo@descaccess}%
8201   \def\@glo@shortaccess{\relax}%
8202   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
8203   \def\@glo@longaccess{\relax}%
8204   \def\@glo@longpluralaccess{\@glo@longaccess}%
8205 }

```

Add to the end hook:

```

8206 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
8207 \renewcommand*{\@newglossaryentryposthook}{%
8208   \@gls@oldnewglossaryentryposthook

```

Store the access information:

```

8209   \expandafter
8210     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
8211     \@glo@access}%
8212   \expandafter
8213     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
8214     \@glo@textaccess}%
8215   \expandafter

```

```

8216 \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
8217 \@glo@firstaccess}%
8218 \expandafter
8219 \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
8220 \@glo@pluralaccess}%
8221 \expandafter
8222 \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
8223 \@glo@firstpluralaccess}%
8224 \expandafter
8225 \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
8226 \@glo@symbolaccess}%
8227 \expandafter
8228 \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
8229 \@glo@symbolpluralaccess}%
8230 \expandafter
8231 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
8232 \@glo@descaccess}%
8233 \expandafter
8234 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
8235 \@glo@descpluralaccess}%
8236 \expandafter
8237 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
8238 \@glo@shortaccess}%
8239 \expandafter
8240 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
8241 \@glo@shortpluralaccess}%
8242 \expandafter
8243 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
8244 \@glo@longaccess}%
8245 \expandafter
8246 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
8247 \@glo@longpluralaccess}%
8248 }

```

6.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

8249 \newcommand*{\glsentryaccess}[1]{%
8250 \csname glo@#1@access\endcsname
8251 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

8252 \newcommand*{\glsentrytextaccess}[1]{%
8253 \csname glo@#1@textaccess\endcsname
8254 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

8255 \newcommand*{\glsentryfirstaccess}[1]{%
8256 \csname glo@#1@firstaccess\endcsname

```

8257 }

\glentrypluralaccess Get the value of the pluralaccess key for the entry with the given label:

```

8258 \newcommand*{\glentrypluralaccess}[1]{%
8259   \csname glo@#1@pluralaccess\endcsname
8260 }
```

\glentryfirstpluralaccess Get the value of the firstpluralaccess key for the entry with the given label:

```

8261 \newcommand*{\glentryfirstpluralaccess}[1]{%
8262   \csname glo@#1@firstpluralaccess\endcsname
8263 }
```

\glentrysymbolaccess Get the value of the symbolaccess key for the entry with the given label:

```

8264 \newcommand*{\glentrysymbolaccess}[1]{%
8265   \csname glo@#1@symbolaccess\endcsname
8266 }
```

\glentrysymbolpluralaccess Get the value of the symbolpluralaccess key for the entry with the given label:

```

8267 \newcommand*{\glentrysymbolpluralaccess}[1]{%
8268   \csname glo@#1@symbolpluralaccess\endcsname
8269 }
```

\glentrydescaccess Get the value of the descriptionaccess key for the entry with the given label:

```

8270 \newcommand*{\glentrydescaccess}[1]{%
8271   \csname glo@#1@descaccess\endcsname
8272 }
```

\glentrydescpluralaccess Get the value of the descriptionpluralaccess key for the entry with the given label:

```

8273 \newcommand*{\glentrydescpluralaccess}[1]{%
8274   \csname glo@#1@descaccess\endcsname
8275 }
```

\glentryshortaccess Get the value of the shortaccess key for the entry with the given label:

```

8276 \newcommand*{\glentryshortaccess}[1]{%
8277   \csname glo@#1@shortaccess\endcsname
8278 }
```

\glentryshortpluralaccess Get the value of the shortpluralaccess key for the entry with the given label:

```

8279 \newcommand*{\glentryshortpluralaccess}[1]{%
8280   \csname glo@#1@shortpluralaccess\endcsname
8281 }
```

\glentrylongaccess Get the value of the longaccess key for the entry with the given label:

```

8282 \newcommand*{\glentrylongaccess}[1]{%
8283   \csname glo@#1@longaccess\endcsname
8284 }
```

trylongpluralaccess Get the value of the longpluralaccess key for the entry with the given label:

```
8285 \newcommand*{\glstrylongpluralaccess}[1]{%
8286   \csname glo@#1@longpluralaccess\endcsname
8287 }
```

\glsaccsupp \glsaccsupp{<replacement text>}{<text>}

This can be redefined to use E or Alt instead of ActualText. (I don't have the software to test the E or Alt options.)

```
8288 \newcommand*{\glsaccsupp}[2]{%
8289   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
8290 }
```

\xglsaccsupp Fully expands replacement text before calling \glsaccsupp

```
8291 \newcommand*{\xglsaccsupp}[2]{%
8292   \protected@edef\@gls@replacementtext{#1}%
8293   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
8294 }
```

lsglnameaccessdisplay Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
8295 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
8296   \protected@edef\@glo@access{\glstryaccess{#2}}%
8297   \ifx\@glo@access\@gls@noaccess
8298     #1%
8299   \else
8300     \xglsaccsupp{\@glo@access}{#1}%
8301   \fi
8302 }
```

lsgltextaccessdisplay As above but for the textaccess replacement text.

```
8303 \DeclareRobustCommand*{\glstextaccessdisplay}[2]{%
8304   \protected@edef\@glo@access{\glstrytextaccess{#2}}%
8305   \ifx\@glo@access\@gls@noaccess
8306     #1%
8307   \else
8308     \xglsaccsupp{\@glo@access}{#1}%
8309   \fi
8310 }
```

lsglpluralaccessdisplay As above but for the pluralaccess replacement text.

```
8311 \DeclareRobustCommand*{\glspluralaccessdisplay}[2]{%
8312   \protected@edef\@glo@access{\glstrypluralaccess{#2}}%
8313   \ifx\@glo@access\@gls@noaccess
8314     #1%
8315   \else
8316     \xglsaccsupp{\@glo@access}{#1}%
8317   \fi
8318 }
```

firstaccessdisplay As above but for the firstaccess replacement text.

```
8319 \DeclareRobustCommand*\glsfirstaccessdisplay}[2]{%
8320   \protected@edef\@glo@access{\glsentryfirstaccess{#2}}%
8321   \ifx\@glo@access\@gls@noaccess
8322     #1%
8323   \else
8324     \xglsaccsupp{\@glo@access}{#1}%
8325   \fi
8326 }
```

pluralaccessdisplay As above but for the firstpluralaccess replacement text.

```
8327 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
8328   \protected@edef\@glo@access{\glsentryfirstpluralaccess{#2}}%
8329   \ifx\@glo@access\@gls@noaccess
8330     #1%
8331   \else
8332     \xglsaccsupp{\@glo@access}{#1}%
8333   \fi
8334 }
```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```
8335 \DeclareRobustCommand*\glsymbolaccessdisplay}[2]{%
8336   \protected@edef\@glo@access{\glsentrysymbolaccess{#2}}%
8337   \ifx\@glo@access\@gls@noaccess
8338     #1%
8339   \else
8340     \xglsaccsupp{\@glo@access}{#1}%
8341   \fi
8342 }
```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```
8343 \DeclareRobustCommand*\glsymbolpluralaccessdisplay}[2]{%
8344   \protected@edef\@glo@access{\glsentrysymbolpluralaccess{#2}}%
8345   \ifx\@glo@access\@gls@noaccess
8346     #1%
8347   \else
8348     \xglsaccsupp{\@glo@access}{#1}%
8349   \fi
8350 }
```

captionaccessdisplay As above but for the descriptionaccess replacement text.

```
8351 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
8352   \protected@edef\@glo@access{\glsentrydescaccess{#2}}%
8353   \ifx\@glo@access\@gls@noaccess
8354     #1%
8355   \else
8356     \xglsaccsupp{\@glo@access}{#1}%
8357   \fi
8358 }
```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```
8359 \DeclareRobustCommand*{\glsdescriptionpluralaccessdisplay}[2]{%
8360   \protected@edef\@glo@access{\glsentrydescpluralaccess{#2}}%
8361   \ifx\@glo@access\@gls@noaccess
8362     #1%
8363   \else
8364     \xglsaccsupp{\@glo@access}{#1}%
8365   \fi
8366 }
```

sshortaccessdisplay As above but for the shortaccess replacement text.

```
8367 \DeclareRobustCommand*{\glsshortaccessdisplay}[2]{%
8368   \protected@edef\@glo@access{\glsentryshortaccess{#2}}%
8369   \ifx\@glo@access\@gls@noaccess
8370     #1%
8371   \else
8372     \xglsaccsupp{\@glo@access}{#1}%
8373   \fi
8374 }
```

pluralaccessdisplay As above but for the shortpluralaccess replacement text.

```
8375 \DeclareRobustCommand*{\glsshortpluralaccessdisplay}[2]{%
8376   \protected@edef\@glo@access{\glsentryshortpluralaccess{#2}}%
8377   \ifx\@glo@access\@gls@noaccess
8378     #1%
8379   \else
8380     \xglsaccsupp{\@glo@access}{#1}%
8381   \fi
8382 }
```

lslongaccessdisplay As above but for the longaccess replacement text.

```
8383 \DeclareRobustCommand*{\glslongaccessdisplay}[2]{%
8384   \protected@edef\@glo@access{\glsentrylongaccess{#2}}%
8385   \ifx\@glo@access\@gls@noaccess
8386     #1%
8387   \else
8388     \xglsaccsupp{\@glo@access}{#1}%
8389   \fi
8390 }
```

pluralaccessdisplay As above but for the longpluralaccess replacement text.

```
8391 \DeclareRobustCommand*{\glslongpluralaccessdisplay}[2]{%
8392   \protected@edef\@glo@access{\glsentrylongpluralaccess{#2}}%
8393   \ifx\@glo@access\@gls@noaccess
8394     #1%
8395   \else
8396     \xglsaccsupp{\@glo@access}{#1}%
8397   \fi
8398 }
```


`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

8399 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
8400   \@ifundefined{gls#1accessdisplay}%
8401   {%
8402     \PackageError{glossaries-accsupp}{No accessibility support
8403       for key ‘#1’}{%
8404   }%
8405   {%
8406     \csname gls#1accessdisplay\endcsname{#2}{#3}%
8407   }%
8408 }
```

`ls@default@entryfmt` Redefine the default entry format to use accessibility information

```

8409 \renewcommand*\ls@default@entryfmt}[2]{%
8410   \ifdefempty\glscustomtext
8411   {%
8412     \glsifplural
8413     {%
      Plural form
8414       \glscapscase
8415       {%
        Don't adjust case
8416         \ifglsused\glslabel
8417         {%
          Subsequent use
8418           #2{\glspluralaccessdisplay
8419             {\glsentryplural{\glslabel}}{\glslabel}}%
8420           {\glsdescriptionpluralaccessdisplay
8421             {\glsentrydescplural{\glslabel}}{\glslabel}}%
8422           {\glsymbolpluralaccessdisplay
8423             {\glsentrysymbolplural{\glslabel}}{\glslabel}}
8424           {\glsinsert}%
8425         }%
8426       }%
      First use
8427       #1{\glsfirstpluralaccessdisplay
8428         {\glsentryfirstplural{\glslabel}}{\glslabel}}%
8429       {\glsdescriptionpluralaccessdisplay
8430         {\glsentrydescplural{\glslabel}}{\glslabel}}%
8431       {\glsymbolpluralaccessdisplay
8432         {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
8433       {\glsinsert}%
8434     }%
8435   }%
8436   {%
```

Make first letter upper case

```
8437      \ifglused\glslabel
8438      {%
```

Subsequent use.

```
8439      #2{\glspluralaccessdisplay
8440          {\Glentryplural{\glslabel}}{\glslabel}}%
8441          {\glsdescriptionpluralaccessdisplay
8442          {\glentrydescplural{\glslabel}}{\glslabel}}%
8443          {\glssymbolpluralaccessdisplay
8444          {\glentrysymbolplural{\glslabel}}{\glslabel}}%
8445          {\glinsert}}%
8446      }%
8447      {%
```

First use

```
8448      #1{\glfirstpluralaccessdisplay
8449          {\Glentryfirstplural{\glslabel}}{\glslabel}}%
8450          {\glsdescriptionpluralaccessdisplay
8451          {\glentrydescplural{\glslabel}}{\glslabel}}%
8452          {\glssymbolpluralaccessdisplay
8453          {\glentrysymbolplural{\glslabel}}{\glslabel}}%
8454          {\glinsert}}%
8455      }%
8456      }%
8457      {%
```

Make all upper case

```
8458      \ifglused\glslabel
8459      {%
```

Subsequent use

```
8460      \MakeUppercase{%
8461      #2{\glspluralaccessdisplay
8462          {\glentryplural{\glslabel}}{\glslabel}}%
8463          {\glsdescriptionpluralaccessdisplay
8464          {\glentrydescplural{\glslabel}}{\glslabel}}%
8465          {\glssymbolpluralaccessdisplay
8466          {\glentrysymbolplural{\glslabel}}{\glslabel}}%
8467          {\glinsert}}%
8468      }%
8469      {%
```

First use

```
8470      \MakeUppercase{%
8471      #1{\glfirstpluralaccessdisplay
8472          {\glentryfirstplural{\glslabel}}{\glslabel}}%
8473          {\glsdescriptionpluralaccessdisplay
8474          {\glentrydescplural{\glslabel}}{\glslabel}}%
8475          {\glssymbolpluralaccessdisplay
8476          {\glentrysymbolplural{\glslabel}}{\glslabel}}%
```

```

8477         {\glsinsert}}}%
8478     }%
8479 }%
8480 }%
8481 {%

```

Singular form

```

8482     \glscapscase
8483     {%

```

Don't adjust case

```

8484     \ifglsused\glslabel
8485     {%

```

Subsequent use

```

8486     #2{\glstextaccessdisplay
8487         {\glsentrytext{\glslabel}}{\glslabel}}%
8488     {\glsdescriptionaccessdisplay
8489         {\glsentrydesc{\glslabel}}{\glslabel}}%
8490     {\glssymbolaccessdisplay
8491         {\glsentrysymbol{\glslabel}}{\glslabel}}%
8492     {\glsinsert}}%
8493 }%
8494 {%

```

First use

```

8495     #1{\glsfirstaccessdisplay
8496         {\glsentryfirst{\glslabel}}{\glslabel}}%
8497     {\glsdescriptionaccessdisplay
8498         {\glsentrydesc{\glslabel}}{\glslabel}}%
8499     {\glssymbolaccessdisplay
8500         {\glsentrysymbol{\glslabel}}{\glslabel}}%
8501     {\glsinsert}}%
8502 }%
8503 }%
8504 {%

```

Make first letter upper case

```

8505     \ifglsused\glslabel
8506     {%

```

Subsequent use

```

8507     #2{\glstextaccessdisplay
8508         {\Glsentrytext{\glslabel}}{\glslabel}}%
8509     {\glsdescriptionaccessdisplay
8510         {\glsentrydesc{\glslabel}}{\glslabel}}%
8511     {\glssymbolaccessdisplay
8512         {\glsentrysymbol{\glslabel}}{\glslabel}}%
8513     {\glsinsert}}%
8514 }%
8515 {%

```

First use

```

8516      #1{\glsfirstaccessdisplay
8517          {\Glsentryfirst{\glslabel}}{\glslabel}}%
8518          {\glsdescriptionaccessdisplay
8519          {\Glsentrydesc{\glslabel}}{\glslabel}}%
8520          {\glssymbolaccessdisplay
8521          {\Glsentrysymbol{\glslabel}}{\glslabel}}%
8522          {\glsinsert}}%
8523      }%
8524  }%
8525  {%

```

Make all upper case

```

8526      \ifglsused\glslabel
8527      {%

```

Subsequent use

```

8528      \MakeUppercase{%
8529      #2{\glstextaccessdisplay
8530          {\Glsentrytext{\glslabel}}{\glslabel}}%
8531          {\glsdescriptionaccessdisplay
8532          {\Glsentrydesc{\glslabel}}{\glslabel}}%
8533          {\glssymbolaccessdisplay
8534          {\Glsentrysymbol{\glslabel}}{\glslabel}}%
8535          {\glsinsert}}%
8536      }%
8537  {%

```

First use

```

8538      \MakeUppercase{%
8539      #1{\glsfirstaccessdisplay
8540          {\Glsentryfirst{\glslabel}}{\glslabel}}%
8541          {\glsdescriptionaccessdisplay
8542          {\Glsentrydesc{\glslabel}}{\glslabel}}%
8543          {\glssymbolaccessdisplay
8544          {\Glsentrysymbol{\glslabel}}{\glslabel}}%
8545          {\glsinsert}}%
8546      }%
8547  }%
8548  }%
8549  }%
8550  {%

```

Custom text provided in \glsdisp

```

8551      \ifglsused{\glslabel}%
8552      {%

```

Subsequent use

```

8553      #2{\glscustomtext}%
8554      {\glsdescriptionaccessdisplay
8555      {\Glsentrydesc{\glslabel}}{\glslabel}}%

```

```

8556      {\glssymbolaccessdisplay
8557        {\glsentrysymbol{\glslabel}}{\glslabel}}%
8558      {\glsinsert}%
8559    }%
8560  {%

```

First use

```

8561      #1{\glscustomtext}%
8562      {\glsdescriptionaccessdisplay
8563        {\glsentrydesc{\glslabel}}{\glslabel}}%
8564      {\glssymbolaccessdisplay
8565        {\glsentrysymbol{\glslabel}}{\glslabel}}%
8566      {\glsinsert}%
8567    }%
8568  }%
8569 }

```

\@acrshort

```

8570 \def\@acrshort#1#2[#3]{%
8571   \glsdoifexists{#2}%
8572   {%
8573     \edef\@glo@type{\glsentrytype{#2}}%

8574     \def\glslabel{#2}%
8575     \let\glsifplural\@secondoftwo
8576     \let\glscapscase\@firstofthree
8577     \let\glsinsert\@empty
8578     \def\glscustomtext{%
8579       \acronymfont{\glsshortaccessdisplay{\glsentryshort{#2}}{#2}}#3%
8580     }%

```

Call \@gls@link

```

8581   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8582 }%
8583 }

```

\@Acrshort

```

8584 \def\@Acrshort#1#2[#3]{%
8585   \glsdoifexists{#2}%
8586   {%
8587     \edef\@glo@type{\glsentrytype{#2}}%

8588     \def\glslabel{#2}%
8589     \let\glsifplural\@secondoftwo
8590     \let\glscapscase\@secondofthree
8591     \let\glsinsert\@empty
8592     \def\glscustomtext{%
8593       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
8594     }%

```

```

Call \@gls@link
8595   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8596   }%
8597 }

```

\@ACRshort

```

8598 \def\@ACRshort#1#2[#3]{%
8599   \glsdoifexists{#2}%
8600   {%
8601     \edef\@glo@type{\glentrytype{#2}}%

8602     \def\glslabel{#2}%
8603     \let\glsifplural\@secondoftwo
8604     \let\glscapscase\@thirdofthree
8605     \let\glsinsert\@empty
8606     \def\glscustomtext{%
8607       \acronymfont{\glsshortaccessdisplay
8608         {\MakeUppercase{\glentryshort{#2}}}{#2}}#3%
8609     }%

```

```

Call \@gls@link
8610   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8611   }%
8612 }

```

\@acrlong

```

8613 \def\@acrlong#1#2[#3]{%
8614   \glsdoifexists{#2}%
8615   {%
8616     \edef\@glo@type{\glentrytype{#2}}%

8617     \def\glslabel{#2}%
8618     \let\glsifplural\@secondoftwo
8619     \let\glscapscase\@firstofthree
8620     \let\glsinsert\@empty
8621     \def\glscustomtext{%
8622       \acronymfont{\glslongaccessdisplay{\glentrylong{#2}}{#2}}#3%
8623     }%

```

```

Call \@gls@link
8624   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8625   }%
8626 }

```

\@Acrlong

```

8627 \def\@Acrlong#1#2[#3]{%
8628   \glsdoifexists{#2}%
8629   {%
8630     \edef\@glo@type{\glentrytype{#2}}%

```

```

8631 \def\glslabel{#2}%
8632 \let\glsifplural\@secondoftwo
8633 \let\glscapscase\@firstofthree
8634 \let\glsinsert\@empty
8635 \def\glscustomtext{%
8636 \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
8637 }%

Call \@gls@link
8638 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8639 }%
8640 }

```

\@ACRlong

```

8641 \def\@ACRlong#1#2[#3]{%
8642 \glsdoifexists{#2}%
8643 {%
8644 \edef\@glo@type{\glsentrytype{#2}}%

8645 \def\glslabel{#2}%
8646 \let\glsifplural\@secondoftwo
8647 \let\glscapscase\@firstofthree
8648 \let\glsinsert\@empty
8649 \def\glscustomtext{%
8650 \acronymfont{\glslongaccessdisplay{%
8651 \MakeUppercase{\glsentrylong{#2}}}{#2}}#3}%
8652 }%

Call \@gls@link
8653 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
8654 }%
8655 }

```

6.3 Displaying the Glossary

We need to redefine the way the glossary entries are formatted to include the accessibility support. The predefined glossary styles use `\glossentryname`, `\glossentrydesc` and `\glossentrysymbol`, but we need to provide compatibility with earlier versions in case users have defined their own styles using `\accsuppglossaryentryfield` and `\accsuppglossarysubentryfield`.

Now redefine `\glossentryname`, `\glossentrydesc` and `\glossentrysymbol` etc so they use the accessibility stuff.

```

8656 \renewcommand*{\glossentryname}[1]{%
8657 \glsdoifexists{#1}%
8658 {%
8659 \glsnamefont{\glsnameaccessdisplay{\glsentryname{#1}}{#1}}%
8660 }%
8661 }

```

```

8662 \renewcommand*{\glossentryname}[1]{%
8663   \glsdoifexists{#1}%
8664   {%
8665     \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
8666   }%
8667 }

8668 \renewcommand*{\glossentrydesc}[1]{%
8669   \glsdoifexists{#1}%
8670   {%
8671     \glsdescriptionaccessdisplay{\glsentrydesc{#1}}{#1}%
8672   }%
8673 }

8674 \renewcommand*{\Glossentrydesc}[1]{%
8675   \glsdoifexists{#1}%
8676   {%
8677     \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
8678   }%
8679 }

8680 \renewcommand*{\glossentrysymbol}[1]{%
8681   \glsdoifexists{#1}%
8682   {%
8683     \glssymbolaccessdisplay{\glsentrysymbol{#1}}{#1}%
8684   }%
8685 }

8686 \renewcommand*{\Glossentrysymbol}[1]{%
8687   \glsdoifexists{#1}%
8688   {%
8689     \glssymbolaccessdisplay{\Glsentrysymbol{#1}}{#1}%
8690   }%
8691 }

```

pglossaryentryfield

```

8692 \newcommand*{\accsuppglossaryentryfield}[5]{%
8693   \glossaryentryfield{#1}%
8694   {\glsnameaccessdisplay{#2}{#1}}%
8695   {\glsdescriptionaccessdisplay{#3}{#1}}%
8696   {\glssymbolaccessdisplay{#4}{#1}}{#5}%
8697 }

```

glossarysubentryfield

```

8698 \newcommand*{\accsuppglossarysubentryfield}[6]{%
8699   \glossarysubentryfield{#1}{#2}%
8700   {\glsnameaccessdisplay{#3}{#2}}%
8701   {\glsdescriptionaccessdisplay{#4}{#2}}%
8702   {\glssymbolaccessdisplay{#5}{#2}}{#6}%
8703 }

```


6.4 Acronyms

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```

8704 \renewcommand*{\newacronymhook}{%
8705   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
8706     \the\glskeylisttok}%
8707   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
8708 }
```

`defaultNewAcronymDef` Modify default style to use access text:

```

8709 \renewcommand*{\DefaultNewAcronymDef}{%
8710   \edef\@do@newglossaryentry{%
8711     \noexpand\newglossaryentry{\the\glslabeltok}%
8712     {%
8713       type=\acronymtype,%
8714       name={\the\glsshorttok},%
8715       description={\the\glslongtok},%
8716       descriptionaccess=\relax,
8717       text={\the\glsshorttok},%
8718       access={\noexpand\@glo@textaccess},%
8719       sort={\the\glsshorttok},%
8720       short={\the\glsshorttok},%
8721       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8722       shortaccess={\the\glslongtok},%
8723       long={\the\glslongtok},%
8724       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8725       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8726       first={\noexpand\glslongaccessdisplay
8727         {\the\glslongtok}{\the\glslabeltok}\space
8728         (\noexpand\glsshortaccessdisplay
8729           {\the\glsshorttok}{\the\glslabeltok})},%
8730       plural={\the\glsshorttok\acrpluralsuffix},%
8731       firstplural={\noexpand\glslongpluralaccessdisplay
8732         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
8733         (\noexpand\glsshortpluralaccessdisplay
8734           {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
8735       firstaccess=\relax,
8736       firstpluralaccess=\relax,
8737       textaccess={\noexpand\@glo@shortaccess},%
8738       \the\glskeylisttok
8739     }%
8740   }%
8741   \@do@newglossaryentry
8742 }
```

`footnoteNewAcronymDef`

```

8743 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
8744   \edef\@do@newglossaryentry{%
```

```

8745 \noexpand\newglossaryentry{\the\glslabeltok}%
8746 {%
8747   type=\acronymtype,%
8748   name={\noexpand\acronymfont{\the\glsshorttok}},%
8749   sort={\the\glsshorttok},%
8750   text={\the\glsshorttok},%
8751   short={\the\glsshorttok},%
8752   shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8753   shortaccess={\the\glslongtok},%
8754   long={\the\glslongtok},%
8755   longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8756   access={\noexpand\@glo@textaccess},%
8757   plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8758   symbol={\the\glslongtok},%
8759   symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8760   firstpluralaccess=\relax,
8761   textaccess={\noexpand\@glo@shortaccess},%
8762   \the\glskeylisttok
8763 }%
8764 }%
8765 \@do@newglossaryentry
8766 }

```

ptionNewAcronymDef

```

8767 \renewcommand*{\DescriptionNewAcronymDef}{%
8768   \edef\@do@newglossaryentry{%
8769     \noexpand\newglossaryentry{\the\glslabeltok}%
8770     {%
8771       type=\acronymtype,%
8772       name={\noexpand
8773         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
8774       access={\noexpand\@glo@textaccess},%
8775       sort={\the\glsshorttok},%
8776       short={\the\glsshorttok},%
8777       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8778       shortaccess={\the\glslongtok},%
8779       long={\the\glslongtok},%
8780       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8781       first={\the\glslongtok},%
8782       firstaccess=\relax,
8783       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8784       text={\the\glsshorttok},%
8785       textaccess={\the\glslongtok},%
8786       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8787       symbol={\noexpand\@glo@text},%
8788       symbolaccess={\noexpand\@glo@textaccess},%
8789       symbolplural={\noexpand\@glo@plural},%
8790       firstpluralaccess=\relax,
8791       textaccess={\noexpand\@glo@shortaccess},%

```

```

8792     \the\glskeylisttok}%
8793 }%
8794 \@do@newglossaryentry
8795 }

```

otnoteNewAcronymDef

```

8796 \renewcommand*{\FootnoteNewAcronymDef}{%
8797   \edef\@do@newglossaryentry{%
8798     \noexpand\newglossaryentry{\the\glslabeltok}%
8799     {%
8800       type=\acronymtype,%
8801       name={\noexpand\acronymfont{\the\glsshorttok}},%
8802       sort={\the\glsshorttok},%
8803       text={\the\glsshorttok},%
8804       textaccess={\the\glslongtok},%
8805       access={\noexpand\@glo@textaccess},%
8806       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8807       short={\the\glsshorttok},%
8808       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8809       long={\the\glslongtok},%
8810       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8811       description={\the\glslongtok},%
8812       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8813       \the\glskeylisttok
8814     }%
8815   }%
8816   \@do@newglossaryentry
8817 }

```

\SmallNewAcronymDef

```

8818 \renewcommand*{\SmallNewAcronymDef}{%
8819   \edef\@do@newglossaryentry{%
8820     \noexpand\newglossaryentry{\the\glslabeltok}%
8821     {%
8822       type=\acronymtype,%
8823       name={\noexpand\acronymfont{\the\glsshorttok}},%
8824       access={\noexpand\@glo@symbolaccess},%
8825       sort={\the\glsshorttok},%
8826       short={\the\glsshorttok},%
8827       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8828       shortaccess={\the\glslongtok},%
8829       long={\the\glslongtok},%
8830       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
8831       text={\noexpand\@glo@short},%
8832       textaccess={\noexpand\@glo@shortaccess},%
8833       plural={\noexpand\@glo@shortpl},%
8834       first={\the\glslongtok},%
8835       firstaccess=\relax,%
8836       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%

```

```

8837     description={\noexpand\@glo@first},%
8838     descriptionplural={\noexpand\@glo@firstplural},%
8839     symbol={\the\glsshorttok},%
8840     symbolaccess={\the\glslongtok},%
8841     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
8842     \the\glskeylisttok
8843   }%
8844 }%
8845 \@do@newglossaryentry
8846 }

```

The following are kept for compatibility with versions before 3.0:

\glsshortaccesskey

```

8847 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

```

\glsshortpluralaccesskey

```

8848 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

```

\glslongaccesskey

```

8849 \newcommand*{\glslongaccesskey}{\glslongkey access}%

```

\glslongpluralaccesskey

```

8850 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

6.5 Debugging Commands

\showglonameaccess

```

8851 \newcommand*{\showglonameaccess}[1]{%
8852   \expandafter\show\csname glo@#1@textaccess\endcsname
8853 }

```

\showglotextaccess

```

8854 \newcommand*{\showglotextaccess}[1]{%
8855   \expandafter\show\csname glo@#1@textaccess\endcsname
8856 }

```

\showglopluralaccess

```

8857 \newcommand*{\showglopluralaccess}[1]{%
8858   \expandafter\show\csname glo@#1@pluralaccess\endcsname
8859 }

```

\showglofirstaccess

```

8860 \newcommand*{\showglofirstaccess}[1]{%
8861   \expandafter\show\csname glo@#1@firstaccess\endcsname
8862 }

```

lofirstpluralaccess

```
8863 \newcommand*{\showglofirstpluralaccess}[1]{%
8864   \expandafter\show\csname glo@#1@firstpluralaccess\endcsname
8865 }
```

showglosymbolaccess

```
8866 \newcommand*{\showglosymbolaccess}[1]{%
8867   \expandafter\show\csname glo@#1@symbolaccess\endcsname
8868 }
```

osymbolpluralaccess

```
8869 \newcommand*{\showglosymbolpluralaccess}[1]{%
8870   \expandafter\show\csname glo@#1@symbolpluralaccess\endcsname
8871 }
```

\showglodescaccess

```
8872 \newcommand*{\showglodescaccess}[1]{%
8873   \expandafter\show\csname glo@#1@descaccess\endcsname
8874 }
```

glodescpluralaccess

```
8875 \newcommand*{\showglodescpluralaccess}[1]{%
8876   \expandafter\show\csname glo@#1@descpluralaccess\endcsname
8877 }
```

\showgloshortaccess

```
8878 \newcommand*{\showgloshortaccess}[1]{%
8879   \expandafter\show\csname glo@#1@shortaccess\endcsname
8880 }
```

loshortpluralaccess

```
8881 \newcommand*{\showgloshortpluralaccess}[1]{%
8882   \expandafter\show\csname glo@#1@shortpluralaccess\endcsname
8883 }
```

\showglolongaccess

```
8884 \newcommand*{\showglolongaccess}[1]{%
8885   \expandafter\show\csname glo@#1@longaccess\endcsname
8886 }
```

glolongpluralaccess

```
8887 \newcommand*{\showglolongpluralaccess}[1]{%
8888   \expandafter\show\csname glo@#1@longpluralaccess\endcsname
8889 }
```

7 Multi-Lingual Support

Many thanks to everyone who contributed to the translations both via email and on comp.text.tex.

7.1 Babel Captions

Define captions if multi-lingual support is required, but the package is not loaded.

```
8890 \NeedsTeXFormat{LaTeX2e}
8891 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]
```

English:

```
8892 \@ifundefined{captionsenglish}{\}{%
8893   \addto\captionsenglish{%
8894     \renewcommand*{\glossaryname}{Glossary}%
8895     \renewcommand*{\acronymname}{Acronyms}%
8896     \renewcommand*{\entryname}{Notation}%
8897     \renewcommand*{\descriptionname}{Description}%
8898     \renewcommand*{\symbolname}{Symbol}%
8899     \renewcommand*{\pagelistname}{Page List}%
8900     \renewcommand*{\glssymbolsgroupname}{Symbols}%
8901     \renewcommand*{\glsnumbersgroupname}{Numbers}%
8902 }%
8903 }
8904 \@ifundefined{captionsamerican}{\}{%
8905   \addto\captionsamerican{%
8906     \renewcommand*{\glossaryname}{Glossary}%
8907     \renewcommand*{\acronymname}{Acronyms}%
8908     \renewcommand*{\entryname}{Notation}%
8909     \renewcommand*{\descriptionname}{Description}%
8910     \renewcommand*{\symbolname}{Symbol}%
8911     \renewcommand*{\pagelistname}{Page List}%
8912     \renewcommand*{\glssymbolsgroupname}{Symbols}%
8913     \renewcommand*{\glsnumbersgroupname}{Numbers}%
8914 }%
8915 }
8916 \@ifundefined{captionsaustralian}{\}{%
8917   \addto\captionsaustralian{%
8918     \renewcommand*{\glossaryname}{Glossary}%
8919     \renewcommand*{\acronymname}{Acronyms}%
8920     \renewcommand*{\entryname}{Notation}%
8921     \renewcommand*{\descriptionname}{Description}%
8922     \renewcommand*{\symbolname}{Symbol}%
8923     \renewcommand*{\pagelistname}{Page List}%
8924     \renewcommand*{\glssymbolsgroupname}{Symbols}%
8925     \renewcommand*{\glsnumbersgroupname}{Numbers}%
8926 }%
8927 }
8928 \@ifundefined{captionsbritish}{\}{%
8929   \addto\captionsbritish{%
8930     \renewcommand*{\glossaryname}{Glossary}%
8931     \renewcommand*{\acronymname}{Acronyms}%
8932     \renewcommand*{\entryname}{Notation}%
8933     \renewcommand*{\descriptionname}{Description}%
```

```

8934 \renewcommand*{\symbolname}{Symbol}%
8935 \renewcommand*{\pagelistname}{Page List}%
8936 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8937 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8938 }}%
8939 \@ifundefined{captionscanadian}{-}{%
8940 \addto\captionscanadian{%
8941 \renewcommand*{\glossaryname}{Glossary}%
8942 \renewcommand*{\acronymname}{Acronyms}%
8943 \renewcommand*{\entryname}{Notation}%
8944 \renewcommand*{\descriptionname}{Description}%
8945 \renewcommand*{\symbolname}{Symbol}%
8946 \renewcommand*{\pagelistname}{Page List}%
8947 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8948 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8949 }}%
8950 }
8951 \@ifundefined{captionsnewzealand}{-}{%
8952 \addto\captionsnewzealand{%
8953 \renewcommand*{\glossaryname}{Glossary}%
8954 \renewcommand*{\acronymname}{Acronyms}%
8955 \renewcommand*{\entryname}{Notation}%
8956 \renewcommand*{\descriptionname}{Description}%
8957 \renewcommand*{\symbolname}{Symbol}%
8958 \renewcommand*{\pagelistname}{Page List}%
8959 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8960 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8961 }}%
8962 }
8963 \@ifundefined{captionsUKenglish}{-}{%
8964 \addto\captionsUKenglish{%
8965 \renewcommand*{\glossaryname}{Glossary}%
8966 \renewcommand*{\acronymname}{Acronyms}%
8967 \renewcommand*{\entryname}{Notation}%
8968 \renewcommand*{\descriptionname}{Description}%
8969 \renewcommand*{\symbolname}{Symbol}%
8970 \renewcommand*{\pagelistname}{Page List}%
8971 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8972 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8973 }}%
8974 }
8975 \@ifundefined{captionsUSenglish}{-}{%
8976 \addto\captionsUSenglish{%
8977 \renewcommand*{\glossaryname}{Glossary}%
8978 \renewcommand*{\acronymname}{Acronyms}%
8979 \renewcommand*{\entryname}{Notation}%
8980 \renewcommand*{\descriptionname}{Description}%
8981 \renewcommand*{\symbolname}{Symbol}%
8982 \renewcommand*{\pagelistname}{Page List}%

```

```

8983 \renewcommand*{\glssymbolsgroupname}{Symbols}%
8984 \renewcommand*{\glsnumbersgroupname}{Numbers}%
8985 }%
8986 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

8987 \@ifundefined{captionsgerman}{}{%
8988 \addto\captionsgerman{%
8989 \renewcommand*{\glossaryname}{Glossar}%
8990 \renewcommand*{\acronymname}{Akronyme}%
8991 \renewcommand*{\entryname}{Bezeichnung}%
8992 \renewcommand*{\descriptionname}{Beschreibung}%
8993 \renewcommand*{\symbolname}{Symbol}%
8994 \renewcommand*{\pagelistname}{Seiten}%
8995 \renewcommand*{\glssymbolsgroupname}{Symbole}%
8996 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
8997 }

```

ngerman is identical to German:

```

8998 \@ifundefined{captionsgerman}{}{%
8999 \addto\captionsgerman{%
9000 \renewcommand*{\glossaryname}{Glossar}%
9001 \renewcommand*{\acronymname}{Akronyme}%
9002 \renewcommand*{\entryname}{Bezeichnung}%
9003 \renewcommand*{\descriptionname}{Beschreibung}%
9004 \renewcommand*{\symbolname}{Symbol}%
9005 \renewcommand*{\pagelistname}{Seiten}%
9006 \renewcommand*{\glssymbolsgroupname}{Symbole}%
9007 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
9008 }

```

Italian:

```

9009 \@ifundefined{captionssitalian}{}{%
9010 \addto\captionssitalian{%
9011 \renewcommand*{\glossaryname}{Glossario}%
9012 \renewcommand*{\acronymname}{Acronimi}%
9013 \renewcommand*{\entryname}{Nomenclatura}%
9014 \renewcommand*{\descriptionname}{Descrizione}%
9015 \renewcommand*{\symbolname}{Simbolo}%
9016 \renewcommand*{\pagelistname}{Elenco delle pagine}%
9017 \renewcommand*{\glssymbolsgroupname}{Simboli}%
9018 \renewcommand*{\glsnumbersgroupname}{Numeri}}
9019 }

```

Dutch:

```

9020 \@ifundefined{captionsdutch}{}{%
9021 \addto\captionsdutch{%
9022 \renewcommand*{\glossaryname}{Woordenlijst}%
9023 \renewcommand*{\acronymname}{Acroniemen}%
9024 \renewcommand*{\entryname}{Benaming}%

```



```

9025 \renewcommand*{\descriptionname}{Beschrijving}%
9026 \renewcommand*{\symbolname}{Symbool}%
9027 \renewcommand*{\pagelistname}{Pagina's}%
9028 \renewcommand*{\glssymbolsgroupname}{Symbolen}%
9029 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
9030 }

```

Spanish:

```

9031 \@ifundefined{captionsspanish}{%}{%
9032 \addto\captionsspanish{%
9033 \renewcommand*{\glossaryname}{Glosario}%
9034 \renewcommand*{\acronymname}{Siglas}%
9035 \renewcommand*{\entryname}{Entrada}%
9036 \renewcommand*{\descriptionname}{Descripci\`on}%
9037 \renewcommand*{\symbolname}{S\`{i}mbolo}%
9038 \renewcommand*{\pagelistname}{Lista de p\`aginas}%
9039 \renewcommand*{\glssymbolsgroupname}{S\`{i}mbolos}%
9040 \renewcommand*{\glsnumbersgroupname}{N\`umeros}}
9041 }

```

French:

```

9042 \@ifundefined{captionsfrench}{%}{%
9043 \addto\captionsfrench{%
9044 \renewcommand*{\glossaryname}{Glossaire}%
9045 \renewcommand*{\acronymname}{Acronymes}%
9046 \renewcommand*{\entryname}{Terme}%
9047 \renewcommand*{\descriptionname}{Description}%
9048 \renewcommand*{\symbolname}{Symbole}%
9049 \renewcommand*{\pagelistname}{Pages}%
9050 \renewcommand*{\glssymbolsgroupname}{Symboles}%
9051 \renewcommand*{\glsnumbersgroupname}{Nombres}}
9052 }
9053 \@ifundefined{captionsfrenchb}{%}{%
9054 \addto\captionsfrenchb{%
9055 \renewcommand*{\glossaryname}{Glossaire}%
9056 \renewcommand*{\acronymname}{Acronymes}%
9057 \renewcommand*{\entryname}{Terme}%
9058 \renewcommand*{\descriptionname}{Description}%
9059 \renewcommand*{\symbolname}{Symbole}%
9060 \renewcommand*{\pagelistname}{Pages}%
9061 \renewcommand*{\glssymbolsgroupname}{Symboles}%
9062 \renewcommand*{\glsnumbersgroupname}{Nombres}}
9063 }
9064 \@ifundefined{captionsfrançais}{%}{%
9065 \addto\captionsfrançais{%
9066 \renewcommand*{\glossaryname}{Glossaire}%
9067 \renewcommand*{\acronymname}{Acronymes}%
9068 \renewcommand*{\entryname}{Terme}%
9069 \renewcommand*{\descriptionname}{Description}%
9070 \renewcommand*{\symbolname}{Symbole}%

```

```

9071 \renewcommand*{\pagelistname}{Pages}%
9072 \renewcommand*{\glssymbolsgroupname}{Symboles}%
9073 \renewcommand*{\glsnumbersgroupname}{Nombres}}
9074 }

```

Danish:

```

9075 \@ifundefined{captionsdanish}{}{%
9076 \addto\captionsdanish{%
9077 \renewcommand*{\glossaryname}{Ordliste}%
9078 \renewcommand*{\acronymname}{Akronymer}%
9079 \renewcommand*{\entryname}{Symbolforklaring}%
9080 \renewcommand*{\descriptionname}{Beskrivelse}%
9081 \renewcommand*{\symbolname}{Symbol}%
9082 \renewcommand*{\pagelistname}{Side}%
9083 \renewcommand*{\glssymbolsgroupname}{Symboler}%
9084 \renewcommand*{\glsnumbersgroupname}{Tal}}
9085 }

```

Irish:

```

9086 \@ifundefined{captionsirish}{}{%
9087 \addto\captionsirish{%
9088 \renewcommand*{\glossaryname}{Gluaís}%
9089 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

9090 \renewcommand*{\entryname}{Ciall}%
9091 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

9092 \renewcommand*{\symbolname}{Comhartha}%
9093 \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
9094 \renewcommand*{\pagelistname}{Leathanaigh}%
9095 \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}
9096 }

```

Hungarian:

```

9097 \@ifundefined{captionsmagyar}{}{%
9098 \addto\captionsmagyar{%
9099 \renewcommand*{\glossaryname}{Sz\ 'ojegyz\ 'ek}%
9100 \renewcommand*{\acronymname}{Bet\ H\ uszavak}%
9101 \renewcommand*{\entryname}{Kifejez\ 'es}%
9102 \renewcommand*{\descriptionname}{Magyar\ 'azat}%
9103 \renewcommand*{\symbolname}{Jel\ "ol\ 'es}%
9104 \renewcommand*{\pagelistname}{Oldalsz\ 'am}%
9105 \renewcommand*{\glssymbolsgroupname}{Jelek}%
9106 \renewcommand*{\glsnumbersgroupname}{Sz\ 'amjegyek}%
9107 }
9108 }
9109 \@ifundefined{captionshungarian}{}{%

```

```

9110 \addto\captionshungarian{%
9111   \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
9112   \renewcommand*{\acronymname}{Bet\H uszavak}%
9113   \renewcommand*{\entryname}{Kifejez\`es}%
9114   \renewcommand*{\descriptionname}{Magyar\`azat}%
9115   \renewcommand*{\symbolname}{Jel\`ol\`es}%
9116   \renewcommand*{\pagelistname}{Oldalsz\`am}%
9117   \renewcommand*{\glssymbolsgroupname}{Jelek}%
9118   \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
9119 }
9120 }

Polish

9121 \@ifundefined{captionspolish}{}{%
9122   \addto\captionspolish{%
9123     \renewcommand*{\glossaryname}{S{l}ownik termin\`ow}%
9124     \renewcommand*{\acronymname}{Skr\`ot}%
9125     \renewcommand*{\entryname}{Termin}%
9126     \renewcommand*{\descriptionname}{Opis}%
9127     \renewcommand*{\symbolname}{Symbol}%
9128     \renewcommand*{\pagelistname}{Strony}%
9129     \renewcommand*{\glssymbolsgroupname}{Symbole}%
9130     \renewcommand*{\glsnumbersgroupname}{Liczby}}
9131 }

Brazilian

9132 \@ifundefined{captionsbrazil}{}{%
9133   \addto\captionsbrazil{%
9134     \renewcommand*{\glossaryname}{Gloss\`ario}%
9135     \renewcommand*{\acronymname}{Siglas}%
9136     \renewcommand*{\entryname}{Nota\c c\~ao}%
9137     \renewcommand*{\descriptionname}{Descri\c c\~ao}%
9138     \renewcommand*{\symbolname}{S\`imbolo}%
9139     \renewcommand*{\pagelistname}{Lista de P\`aginas}%
9140     \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%
9141     \renewcommand*{\glsnumbersgroupname}{N\`umeros}%
9142   }%
9143 }

7.2 Polyglossia Captions

9144 \NeedsTeXFormat{LaTeX2e}
9145 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

English:

9146 \@ifundefined{captionsenglish}{}{%
9147   \expandafter\toks@\expandafter{\captionsenglish
9148     \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
9149     \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
9150     \renewcommand*{\entryname}{\textenglish{Notation}}%
9151     \renewcommand*{\descriptionname}{\textenglish{Description}}}%

```

```

9152 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
9153 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
9154 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
9155 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
9156 }%
9157 \edef\captionseenglish{\the\toks@}%
9158 }

```

German:

```

9159 \@ifundefined{captionsgerman}{}{%
9160 \expandafter\toks@\expandafter{\captionsgerman
9161 \renewcommand*{\glossaryname}{\textgerman{Glossar}}%
9162 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
9163 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
9164 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
9165 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
9166 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
9167 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
9168 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
9169 }%
9170 \edef\captionsgerman{\the\toks@}%
9171 }

```

Italian:

```

9172 \@ifundefined{captionssitalian}{}{%
9173 \expandafter\toks@\expandafter{\captionssitalian
9174 \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
9175 \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
9176 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
9177 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
9178 \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
9179 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
9180 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
9181 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
9182 }%
9183 \edef\captionssitalian{\the\toks@}%
9184 }

```

Dutch:

```

9185 \@ifundefined{captionsdutch}{}{%
9186 \expandafter\toks@\expandafter{\captionsdutch
9187 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
9188 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
9189 \renewcommand*{\entryname}{\textdutch{Benaming}}%
9190 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
9191 \renewcommand*{\symbolname}{\textdutch{Symbool}}%
9192 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
9193 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
9194 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
9195 }%
9196 \edef\captionsdutch{\the\toks@}%

```

9197 }

Spanish:

```
9198 \@ifundefined{captionsspanish}{}{%
9199   \expandafter\toks@\expandafter{\captionsspanish
9200     \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
9201     \renewcommand*{\acronymname}{\textspanish{Siglas}}%
9202     \renewcommand*{\entryname}{\textspanish{Entrada}}%
9203     \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}%
9204     \renewcommand*{\symbolname}{\textspanish{S'\i mbolo}}%
9205     \renewcommand*{\pagelistname}{\textspanish{Lista de p'aginas}}%
9206     \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i mbolos}}%
9207     \renewcommand*{\glsnumbersgroupname}{\textspanish{N'umeros}}%
9208   }%
9209   \edef\captionsspanish{\the\toks@}%
9210 }
```

French:

```
9211 \@ifundefined{captionsfrench}{}{%
9212   \expandafter\toks@\expandafter{\captionsfrench
9213     \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
9214     \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
9215     \renewcommand*{\entryname}{\textfrench{Terme}}%
9216     \renewcommand*{\descriptionname}{\textfrench{Description}}%
9217     \renewcommand*{\symbolname}{\textfrench{Symbole}}%
9218     \renewcommand*{\pagelistname}{\textfrench{Pages}}%
9219     \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
9220     \renewcommand*{\glsnumbersgroupname}{\textfrench{Nombres}}%
9221   }%
9222   \edef\captionsfrench{\the\toks@}%
9223 }
```

Danish:

```
9224 \@ifundefined{captionsdanish}{}{%
9225   \expandafter\toks@\expandafter{\captionsdanish
9226     \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%
9227     \renewcommand*{\acronymname}{\textdanish{Akronymer}}%
9228     \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}%
9229     \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}%
9230     \renewcommand*{\symbolname}{\textdanish{Symbol}}%
9231     \renewcommand*{\pagelistname}{\textdanish{Side}}%
9232     \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
9233     \renewcommand*{\glsnumbersgroupname}{\textdanish{Tal}}%
9234   }%
9235   \edef\captionsdanish{\the\toks@}%
9236 }
```

Irish:

```
9237 \@ifundefined{captionsirish}{}{%
9238   \expandafter\toks@\expandafter{\captionsirish
9239     \renewcommand*{\glossaryname}{\textirish{Gluais}}%
9240     \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%

```

```

9241 \renewcommand*{\entryname}{\textirish{Ciall}}}%
9242 \renewcommand*{\descriptionname}{\textirish{Tuaisc}}}%
9243 \renewcommand*{\symbolname}{\textirish{Comhartha}}}%
9244 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\'}{\i}}}%
9245 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}}%
9246 \renewcommand*{\glslnumbersgroupname}{\textirish{Uimhreacha}}}%
9247 }%
9248 \edef\captionisirish{\the\toks@}%
9249 }

```

Hungarian:

```

9250 \@ifundefined{captionsmagyar}{}{%
9251 \expandafter\toks@\expandafter{\captionsmagyar
9252 \renewcommand*{\glossaryname}{\textmagyar{Sz\'}ojegyz\'ek}}}%
9253 \renewcommand*{\acronymname}{\textmagyar{Bet\'}H uszavak}}}%
9254 \renewcommand*{\entryname}{\textmagyar{Kifejez\'es}}}%
9255 \renewcommand*{\descriptionname}{\textmagyar{Magyar\'}azat}}}%
9256 \renewcommand*{\symbolname}{\textmagyar{Jel\'ol\'es}}}%
9257 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\'am}}}%
9258 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
9259 \renewcommand*{\glslnumbersgroupname}{\textmagyar{Sz\'amjegyek}}}%
9260 }%
9261 \edef\captionsmagyar{\the\toks@}%
9262 }

```

Polish

```

9263 \@ifundefined{captionspolish}{}{%
9264 \expandafter\toks@\expandafter{\captionspolish
9265 \renewcommand*{\glossaryname}{\textpolish{S\'}ownik termin\'}ow}}}%
9266 \renewcommand*{\acronymname}{\textpolish{Skr\'}ot}}}%
9267 \renewcommand*{\entryname}{\textpolish{Termin}}}%
9268 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
9269 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
9270 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
9271 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%
9272 \renewcommand*{\glslnumbersgroupname}{\textpolish{Liczby}}}%
9273 }%
9274 \edef\captionspolish{\the\toks@}%
9275 }

```

Portugues

```

9276 \@ifundefined{captionsportuges}{}{%
9277 \expandafter\toks@\expandafter{\captionsportuges
9278 \renewcommand*{\glossaryname}{\textportuges{Gloss\'}ario}}}%
9279 \renewcommand*{\acronymname}{\textportuges{Siglas}}}%
9280 \renewcommand*{\entryname}{\textportuges{Nota\'}c c\~ao}}}%
9281 \renewcommand*{\descriptionname}{\textportuges{Descri\'}c c\~ao}}}%
9282 \renewcommand*{\symbolname}{\textportuges{S\'}imbolo}}}%
9283 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'}aginas}}}%
9284 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'}imbolos}}}%
9285 \renewcommand*{\glslnumbersgroupname}{\textportuges{N\'}umeros}}}%

```

```

9286 }%
9287 \edef\captionoportuges{\the\toks0}%
9288 }

```

7.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```

9289 \ProvidesDictionary{glossaries-dictionary}{Brazilian}

```

Provide Brazilian translations:

```

9290 \providetranslation{Glossary}{Gloss\’ario}
9291 \providetranslation{Acronyms}{Siglas}
9292 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
9293 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
9294 \providetranslation{Symbol (glossaries)}{S\’imbolo}
9295 \providetranslation{Page List (glossaries)}{Lista de P\’aginas}
9296 \providetranslation{Symbols (glossaries)}{S\’imbolos}
9297 \providetranslation{Numbers (glossaries)}{N\’umeros}

```

7.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

9298 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

9299 \providetranslation{Glossary}{Ordliste}
9300 \providetranslation{Acronyms}{Akronymer}
9301 \providetranslation{Notation (glossaries)}{Symbolforklaring}
9302 \providetranslation{Description (glossaries)}{Beskrivelse}
9303 \providetranslation{Symbol (glossaries)}{Symbol}
9304 \providetranslation{Page List (glossaries)}{Side}
9305 \providetranslation{Symbols (glossaries)}{Symboler}
9306 \providetranslation{Numbers (glossaries)}{Tal}

```

7.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```

9307 \ProvidesDictionary{glossaries-dictionary}{Dutch}

```

Provide Dutch translations:

```

9308 \providetranslation{Glossary}{Woordenlijst}
9309 \providetranslation{Acronyms}{Acroniemen}
9310 \providetranslation{Notation (glossaries)}{Benaming}
9311 \providetranslation{Description (glossaries)}{Beschrijving}
9312 \providetranslation{Symbol (glossaries)}{Symbool}
9313 \providetranslation{Page List (glossaries)}{Pagina’s}
9314 \providetranslation{Symbols (glossaries)}{Symbolen}
9315 \providetranslation{Numbers (glossaries)}{Cijfers}

```

7.6 English Dictionary

This is a dictionary file provided for use with the package.

```
9316 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
9317 \providetranslation{Glossary}{Glossary}
9318 \providetranslation{Acronyms}{Acronyms}
9319 \providetranslation{Notation (glossaries)}{Notation}
9320 \providetranslation{Description (glossaries)}{Description}
9321 \providetranslation{Symbol (glossaries)}{Symbol}
9322 \providetranslation{Page List (glossaries)}{Page List}
9323 \providetranslation{Symbols (glossaries)}{Symbols}
9324 \providetranslation{Numbers (glossaries)}{Numbers}
```

7.7 French Dictionary

This is a dictionary file provided for use with the package.

```
9325 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
9326 \providetranslation{Glossary}{Glossaire}
9327 \providetranslation{Acronyms}{Acronymes}
9328 \providetranslation{Notation (glossaries)}{Terme}
9329 \providetranslation{Description (glossaries)}{Description}
9330 \providetranslation{Symbol (glossaries)}{Symbole}
9331 \providetranslation{Page List (glossaries)}{Pages}
9332 \providetranslation{Symbols (glossaries)}{Symboles}
9333 \providetranslation{Numbers (glossaries)}{Nombres}
```

7.8 German Dictionary

This is a dictionary file provided for use with the package.

```
9334 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
9335 \providetranslation{Glossary}{Glossar}
9336 \providetranslation{Acronyms}{Akronyme}
9337 \providetranslation{Notation (glossaries)}{Bezeichnung}
9338 \providetranslation{Description (glossaries)}{Beschreibung}
9339 \providetranslation{Symbol (glossaries)}{Symbol}
9340 \providetranslation{Page List (glossaries)}{Seiten}
9341 \providetranslation{Symbols (glossaries)}{Symbole}
9342 \providetranslation{Numbers (glossaries)}{Zahlen}
```

7.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
9343 \ProvidesDictionary{glossaries-dictionary}{Irish}
```


Provide Irish translations:

```
9344 \providetranslation{Glossary}{Gluais}
9345 \providetranslation{Acronyms}{Acrainmneacha}
9346 \providetranslation{Notation (glossaries)}{Ciall}
9347 \providetranslation{Description (glossaries)}{Tuairisc}
9348 \providetranslation{Symbol (glossaries)}{Comhartha}
9349 \providetranslation{Page List (glossaries)}{Leathanaigh}
9350 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
9351 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

7.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
9352 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
9353 \providetranslation{Glossary}{Glossario}
9354 \providetranslation{Acronyms}{Acronimi}
9355 \providetranslation{Notation (glossaries)}{Nomenclatura}
9356 \providetranslation{Description (glossaries)}{Descrizione}
9357 \providetranslation{Symbol (glossaries)}{Simbolo}
9358 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
9359 \providetranslation{Symbols (glossaries)}{Simboli}
9360 \providetranslation{Numbers (glossaries)}{Numeri}
```

7.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
9361 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
9362 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
9363 \providetranslation{Acronyms}{Bet\'H uszavak}
9364 \providetranslation{Notation (glossaries)}{Kifejez\'es}
9365 \providetranslation{Description (glossaries)}{Magyar\'azat}
9366 \providetranslation{Symbol (glossaries)}{Jel\'ol\'es}
9367 \providetranslation{Page List (glossaries)}{Oldalsz\'am}
9368 \providetranslation{Symbols (glossaries)}{Jelek}
9369 \providetranslation{Numbers (glossaries)}{Sz\'amjegyek}
```

7.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
9370 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```
9371 \providetranslation{Glossary}{S{\l}ownik termin\'ow}
9372 \providetranslation{Acronyms}{Skr\'ot}
9373 \providetranslation{Notation (glossaries)}{Termin}
```

```

9374 \providetranslation{Description (glossaries)}{Opis}
9375 \providetranslation{Symbol (glossaries)}{Symbol}
9376 \providetranslation{Page List (glossaries)}{Strony}
9377 \providetranslation{Symbols (glossaries)}{Symbole}
9378 \providetranslation{Numbers (glossaries)}{Liczby}

```

7.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

9379 \ProvidesDictionary{glossaries-dictionary}{Serbian}
9380 \providetranslation{Glossary}{Mali re\ v cnik}
9381 \providetranslation{Acronyms}{Skra\' cenice}
9382 \providetranslation{Notation (glossaries)}{Oznaka}
9383 \providetranslation{Description (glossaries)}{Opis}
9384 \providetranslation{Symbol (glossaries)}{Simbol}
9385 \providetranslation{Page List (glossaries)}{Stranica}
9386 \providetranslation{Symbols (glossaries)}{Simboli}
9387 \providetranslation{Numbers (glossaries)}{Brojevi}

```

7.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```

9388 \ProvidesDictionary{glossaries-dictionary}{Spanish}

```

Provide Spanish translations:

```

9389 \providetranslation{Glossary}{Glosario}
9390 \providetranslation{Acronyms}{Siglas}
9391 \providetranslation{Notation (glossaries)}{Entrada}
9392 \providetranslation{Description (glossaries)}{Descripci\'on}
9393 \providetranslation{Symbol (glossaries)}{S\'mbolo}
9394 \providetranslation{Page List (glossaries)}{Lista de p\'aginas}
9395 \providetranslation{Symbols (glossaries)}{S\'mbolos}
9396 \providetranslation{Numbers (glossaries)}{N\'umeros}

```

Glossary

`makeindex` An indexing application. [9](#), [21](#)

`xindy` An flexible indexing application with multilingual support written in Perl. [9](#), [21](#)

Change History

- 1.01
 General: Added range facility in
 formatkey 83
 \writeist: Added spaces after
 \delimN and \delimR in ist
 file 140
- 1.03
 \makefirstuc: changed 'pro-
 tected@edef to 'def 207
- 1.04
 General: Added \glstextformat 72
- 1.05
 \glossarysection: added
 \@mkboth to \glossarysection
 31
 \gls@defglossaryentry:
 Changed the default value of
 the sort key to just the value of
 the name key 62
 \glsmakefirstuc: new 208
- 1.06
 General: now requires etoolbox . 206
 \capitalisewords: new 208
 \xcapitalisewords: new 208
- 1.07
 \@gls@link: fixed bug caused by
 \theglsentrycounter set-
 ting the page number too soon 81
 \glsadd: fixed bug caused by
 \theglsentrycounter set-
 ting the page number too soon
 137
- 1.08
 General: Added babel support ... 26
 \capitalisewords: made robust
 208
 listgroup: changed listgroup
 style to use \glsgetgrouptitle
 214
 altlistgroup: changed al-
 tlistgroup style to use
 \glsgetgrouptitle 215
 \makefirstuc: made robust ... 207
- 1.1
 \@glossarysection: numbered
 sections and auto label added 32
- \@gls@tmpb: changed \toksdef
 to \newtoks 85
 \@gls@toc: numberline added .. 34
 \@p@glossarysection: num-
 bered sections and auto label
 added 33
 General: Added support for trans-
 lator package 26
 amsgen now loaded (\new@ifnextchar
 needed) 4
 translate: translate option
 added 19
 \setglossarysection: new ... 32
 numberedsection: numbered-
 section package option added . 6
 numberline: numberline option
 added 5
- 1.12
 \@GLSpl: now uses \glsentrydescplural
 and \glsentrysymbolplural
 instead of \glsentrydesc
 and \glsentrysymbol 99
 \@Glspl@: now uses \glsentrydescplural
 and \glsentrysymbolplural
 instead of \glsentrydesc
 and \glsentrysymbol 98
 General: added check for
 \hypertarget separate to
 \hyperlink (memoir de-
 fines \hyperlink but not
 \hypertarget) 91
 descriptionplural: new 50
 \gls@defglossaryentry:
 Changed default first plural to
 be first key with s appended
 (was text key with s appended) 62
 descriptionplural support
 added 62
 symbolplural support added .. 62
 \Glsentrydescplural: New .. 132
 \glsentrydescplural: New .. 132
 \Glsentrysymbolplural: New 133
 \glsentrysymbolplural: New 132
 \SetDescriptionFootnoteAcronymStyle:
 Added \protect before
 \footnote and \glslink . 179

\SetFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 185 symbolplural: new 51	\glssettoctitle: new 25 \printglossary: changed the way the TOC title is set 155
1.13	1.16
General: Add Polish support 299, 302 fixed bug that ignored 3rd pa- rameter 102–114 \ACRfullpl: new 175 \Acrfullpl: new 174 \acrfullpl: new 174 \acrpluralsuffix: New 172 \gls@defglossaryentry: Changed default first value .. 62 Changed default firstplural value 62 Removed restriction on only using \newglossaryentry in the preamble 67 \newacronym: Removed re- striction on only using \newacronym in the preamble 172	\@GLS@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 95 \@GLSpl: Test glossary type is \acronymtype in addition to checking if footnote option has been used 99 \@Gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 94 \@GLspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 98 \@gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 93 \@glsdisp: Test glossary type is \acronymtype in addition to checking if footnote option has been used 101 \@glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 97 \@glstarget: raised the hyper- target so the target text doesn't scroll off the top of the page . 91 \gls@defglossaryentry: Changed def to let 62
1.14	1.17
\@gls@hypergroup: new 210 General: added nonumberlist key to \printglossary 159 added numberedsection key to \printglossary 158 \firstacronymfont: new 175 \glsautoprefix: new 6 \glsnavhyperlink: changed 'edef to 'protected@edef ... 209 \glsnavhypertarget: added write to aux file 209 \glsnavigation: changed to only use labels for groups that are present 210	\@@do@wrglossary: new 150 \@do@seeglossary: new 152 \@glo@storeentry: new 68 \@glossary: changed defini- tion to use \index instead of \@index 149 \@glsdefaultplural: new 54 \@glsdefaultsort: new 54 \@glshypernumber: new 169 \@glsnoname: new 53 \@glsnonextpages: new 159
1.15	
\@gls@link: added \glslabel . 81 General: Added \glssettoctitle 26 \gls@defglossaryentry: check for \@glo@first in descrip- tion 66 check for \@glo@text in sym- bol 66 \gls@hypergroup: new . 210 \glsnavhypertarget: added check if rerun required 209	

<code>\@wrglossary:</code> modified to allow for xindy support	149	<code>\gls@defglossaryentry:</code> Changed default value of sort to <code>\@glsdefaultsort</code>	62
General: added Brazilian dictio- nary	303	moved sort sanitization to <code>\newglossaryentry</code>	66
Added Brazilian support	299	<code>\glstarget:new</code>	162
added xindy support	21	<code>\oldaacronym:new</code>	171
<code>parent:new</code>	52	<code>nolist:new</code>	8
<code>see:new</code>	52	<code>nolong:new</code>	7
<code>\gls@defglossaryentry:</code> added nonumberlist key	63	<code>sort:</code> moved sanitization to <code>\newglossaryentry</code>	50
added parent key	63	<code>nostyles:new</code>	8
added see key	62	<code>nosuper:new</code>	8
Stored main part of entry format when entry is defined	67	<code>notree:new</code>	8
<code>\gls@suffixF:new</code>	29	1.19	
<code>\gls@suffixFF:new</code>	29	<code>\gls@clearpage:new</code>	34
<code>\glshyperlink:new</code>	137	<code>\glsdisp:new</code>	100
<code>\glshypernumber:</code> modified to allow material to be attached to location	169	<code>\SetDescriptionAcronymStyle:</code> changed <code>\acronymfont</code> to use <code>\textsmaller</code> instead of <code>\smaller</code>	183
<code>\glsnavhyperlink:</code> replaced 'hy- perlink to '@glslink	209	<code>\SetDescriptionFootnoteAcronymStyle:</code> changed <code>\acronymfont</code> to use <code>\textsmaller</code> instead of <code>\smaller</code>	179
<code>\glsnavhypertarget:</code> replaced 'hypertarget to '@glstarget	209	<code>\SetFootnoteAcronymStyle:</code> changed <code>\acronymfont</code> to use <code>\textsmaller</code> instead of <code>\smaller</code>	185
<code>\glssee:new</code>	153	<code>\SetSmallAcronymStyle:</code> changed <code>\acronymfont</code> to use <code>\textsmaller</code> instead of <code>\smaller</code>	188
<code>\glsseeformat:new</code>	153	1.2	
<code>\glsSetSuffixF:new</code>	29	General: fixed bug in ngerman captions	296
<code>\glsSetSuffixFF:new</code>	30	2.01	
<code>\ifglsxindy:new</code>	21	<code>\@gls@link:</code> moved <code>\@do@wrglossary</code> before term is displayed to pre- vent unwanted whatsit	82
<code>\istfilename:</code> added xindy sup- port	28	General: added nomain package option	12
<code>\newglossarystyle:</code> made <code>\newglossarystyle long</code>	168	<code>\forallglossaries:</code> replaced <code>\ifthenelse</code> with <code>\ifx</code>	43
<code>\nopostdesc:new</code>	27	<code>\forallglsentries:</code> replaced <code>\ifthenelse</code> with <code>\ifx</code>	43
<code>nonumberlist:new</code>	52	<code>\glsdefmain:new</code>	12
<code>\printglossary:</code> added check to determine if <code>\printglossary</code> is already defined	155	<code>\glsdescwidth:</code> changed <code>\linewidth</code> to <code>\hsize</code>	217, 232
added print language to aux file	155		
order: order package option added	20		
<code>\writeist:</code> added xindy support	140		
1.18			
<code>\@gls@loadlist:new</code>	8		
<code>\@gls@loadlong:new</code>	7		
<code>\@gls@loadsuper:new</code>	7		
<code>\@gls@loadtree:new</code>	8		

<code>\glslistdottedwidth</code> : changed <code>\linewidth</code> to <code>\hsize</code> 216	glossary type has been identified as a list of acronyms 94
<code>\glspagelistwidth</code> : changed <code>\linewidth</code> to <code>\hsize</code> . 217, 232	<code>\@Glspl@</code> : Changed test to check if glossary type has been identified as a list of acronyms ... 98
<code>\writeist</code> : removed item_02 - no such makeindex key 144	<code>\@glossaryentryfield</code> : new .. 67
2.02	<code>\@glossarysubentryfield</code> : new 67
General: Changed Brazil to Brazilian 303	<code>\@gls@</code> : Changed test to check if glossary type has been identified as a list of acronyms 93
false will prevent automatic loading of translator package 23	<code>\@glsacronymlists</code> : new 13
<code>\glossarysection</code> : changed <code>\@mkboth</code> to <code>\glossarymark</code> 31	<code>\@glsdisp</code> : Changed test to check if glossary type has been identified as a list of acronyms .. 101
<code>\gls glossarymark</code> : New 31	<code>\@Glspl@</code> : Changed test to check if glossary type has been identified as a list of acronyms ... 97
<code>\printglossary</code> : suppressed warning globally rather than locally 157	<code>\@newglossaryentryposthook</code> : new 67
2.03	<code>\@newglossaryentryprehook</code> : new 67
<code>\@GLS@</code> : Added check for hyperfirst 95	<code>acronymlists</code> : new 14
<code>\@GLSpl</code> : Added check for hyperfirst 99	<code>\DeclareAcronymList</code> : new ... 13
<code>\@GLs@</code> : Added check for hyperfirst 94	<code>\DefineAcronymSynonyms</code> : new 192
<code>\@Glspl@</code> : Added check for hyperfirst 98	<code>\gls@defglossaryentry</code> : added user1-6 keys 63
<code>\@gls@</code> : Added check for hyperfirst 93	<code>\glsadd</code> : fixed bug that ignored counter 137
<code>\@gls@link</code> : new 81	<code>\Glsentryuseri</code> : new 133
<code>\@gls@link</code> : added <code>\leavevmode</code> 81	<code>\glsentryuseri</code> : new 133
Moved entry existence check to avoid duplicate code 81	<code>\Glsentryuserii</code> : new 134
<code>\@glsdisp</code> : Added check for hyperfirst 101	<code>\glsentryuserii</code> : new 134
<code>\@Glspl@</code> : Added check for hyperfirst 97	<code>\Glsentryuseriii</code> : new 134
<code>\gls glossarymark</code> : Added check to see if it's already defined .. 31	<code>\glsentryuseriii</code> : new 134
<code>hyperfirst</code> : new 19	<code>\Glsentryuseriv</code> : new 134
2.04	<code>\glsentryuseriv</code> : new 134
<code>\@GLS@</code> : Changed test to check if glossary type has been identified as a list of acronyms 95	<code>\Glsentryuseriv</code> : new 134
<code>\@GLSpl</code> : Changed test to check if glossary type has been identified as a list of acronyms 99	<code>\Glsentryuserv</code> : new 134
<code>\@GLs@</code> : Changed test to check if	<code>\glsentryuserv</code> : new 134
	<code>\Glsentryuservi</code> : new 134
	<code>\glsentryuservi</code> : new 134
	<code>\newglossary</code> : added check to determine if <code>\gls@<type>@display</code> and <code>\gls@<type>@displayfirst</code> have been defined. 48
	<code>\SetAcronymLists</code> : new 14
	<code>\SetDefaultAcronymDisplayStyle</code> : new 176

\SetDefaultAcronymStyle:	\@gls@saveentrycounter: new	82
new	\@gls@setupsort@def: new	10
\SetDescriptionAcronymDisplayStyle:	\@gls@setupsort@standard:	
new	new	10
\SetDescriptionDUAAcronymDisplayStyle:	\@gls@setupsort@use: new	11
new	\@gls@xdy@locationlist: new	38
\SetDescriptionFootnoteAcronymDisplayStyle:	\@gls@link: replaced \@ifundefined	
new	with \ifcsundef	91
\SetDUADisplayStyle: new ..	\@gls@nextpages: new	159
\SetFootnoteAcronymDisplayStyle:	\@makeglossary: Added check	
new	for savewrites	146
\SetSmallAcronymDisplayStyle:	\@set@glo@numformat: added	
new	4th argument	83
2.05	\@wrglossary: modified to take	
\@glsdisp: Added closing brace.	into account savewrites	149
Patch provided by Sergiu	\@xdy@attributelist: new	34
Dotenco	General: added prefix to hyperlink	
Removed spurious brace. Patch	170
provided by Sergiu Dotenco	etoolbox now loaded	4
101	replaced \@ifundefined with	
\writeist: Added \string be-	\ifcsundef	24, 80, 158
fore opening and closing	\acrfootnote: new	177
braces. Patch provided by	\ACRfull: added starred version	174
Segiu Dotenco	\Acrfull: added starred version	173
144	\acrfull: added starred version	173
2.06	\ACRfullpl: added starred ver-	
\altnewglossary: new	sion	175
\CustomAcronymFields: new .	\Acrfullpl: added starred ver-	
191	sion	174
\CustomNewAcronymDef: new .	\acrfullpl: added starred ver-	
191	sion	174
\SetCustomDisplayStyle: new	\acrfullpl: added starred ver-	
191	sion	174
\SetCustomStyle: new	\acrfullpl: added starred ver-	
192	sion	174
2.07	\acrfullpl: added starred ver-	
General: glssadd format key	sion	174
stored in \@gls@numberformat	\acrlinkfootnote: new	177
(was mistakenly stored in	\acrnolinkfootnote: new ...	177
\@glo@format)	\addglossarytocaptions: re-	
137	placed \@ifundefined with	
3.0	\ifcsundef	26
\@@do@wrglossary: added check	savewrites: new	22
for hyper location prefix ...	see: added \@glo@seeautonumberlist	
151	52
modified to use new format ..	seeautonumberlist: new	7
150	\glossarysection: replaced	
\@glossarysec: replaced	\@ifundefined with	
\@ifundefined with	\ifcsundef	31
\ifcsundef	\glossarystyle: replaced	
5	\@ifundefined with	
\@do@seeglossary: Sanitize and	\ifcsundef	167
escape cross-referencing in-	\@gls@codepage: replaced	
formation	\@ifundefined with	
152	\ifcsundef	21
\@gls@counterwithin: new		
9		
\@gls@ifinlist: new		
35		
\@gls@link: added \@gls@saveentrycounter		
.....		
82		
added \@gls@setsort		
82		

<code>\gls@defglossaryentry:</code> added	<code>theglossary:</code> replaced <code>\@ifundefined</code>
<code>\@gls@defsort</code> 66	with <code>\ifcsundef</code> 162
added short and long keys 63	<code>short:</code> new 53
replaced <code>\@ifundefined</code> with	<code>shortplural:</code> new 53
<code>\ifcsundef</code> 63	<code>\ifglossaryexists:</code> re-
<code>\gls@docclearpage:</code> replaced	placed <code>\@ifundefined</code> with
<code>\@ifundefined</code> with	<code>\ifcsundef</code> 44
<code>\ifcsundef</code> 33	<code>\ifglentryexists:</code> re-
<code>\glsadd:</code> added <code>\@gls@saveentrycounter</code>	placed <code>\@ifundefined</code> with
..... 138	<code>\ifcsundef</code> 44
<code>\GlsAddXdyCounters:</code> new 35	<code>\istfile:</code> deprecated 148
<code>\glentrycounterlabel:</code> new 161	<code>glossaryentry:</code> new 160
<code>\glentryitem:</code> new 161	<code>glossarysubentry:</code> new 160
<code>\Glsentrylong:</code> new 135	<code>\newglossary:</code> added <code>\@gls@defsortcount</code>
<code>\glentrylong:</code> new 135 48
<code>\Glsentrylongpl:</code> new 135	replaced <code>\@ifundefined</code> with
<code>\glentrylongpl:</code> new 135	<code>\ifcsundef</code> 48
<code>\Glsentryshort:</code> new 135	<code>\newglossaryentry:</code> replaced
<code>\glentryshort:</code> new 134	<code>\DeclareRobustCommand</code>
<code>\Glsentryshortpl:</code> new 135	with <code>\newrobustcmd</code> 56
<code>\glentryshortpl:</code> new 135	<code>\newglossarystyle:</code> re-
<code>\glsgetgrouptitle:</code> re-	placed <code>\@ifundefined</code> with
placed <code>\@ifundefined</code> with	<code>\ifcsundef</code> 168
<code>\ifcsundef</code> 166	<code>entrycounter:</code> new 9
<code>\gls glossarymark:</code> replaced	<code>entrycounterwithin:</code> new 9
<code>\@ifundefined</code> with	<code>\oldacronym:</code> replaced <code>\@ifundefined</code>
<code>\ifcsundef</code> 31	with <code>\ifcsundef</code> 171
<code>\glshyperlink:</code> changed de-	<code>compatible-2.07:</code> compatible-
fault from <code>\glentryname</code> to	2.07 option added 22
<code>\glentrytext</code> 137	<code>long:</code> new 53
<code>\glshypernumber:</code> replaced	<code>longplural:</code> new 53
<code>\@ifundefined</code> with	<code>nonumberlist:</code> now boolean ... 52
<code>\ifcsundef</code> 169	<code>sort:</code> new 9
<code>\glsnumberformat:</code> replaced	<code>counter:</code> replaced <code>\@ifundefined</code>
<code>\@ifundefined</code> with	with <code>\ifcsundef</code> 51
<code>\ifcsundef</code> 30	<code>\printglossary:</code> added
<code>\glsrefentry:</code> new 161	<code>\currentglossary</code> 156
<code>\glsresetsubentrycounter:</code>	added <code>\glsnextpages</code> 156
new 160	make toctitle default to title .. 155
<code>\glsseeitem:</code> hyperlink uses	replaced <code>\@ifundefined</code> with
<code>\glsseeitemformat</code> instead	<code>\ifcsundef</code> 155, 156
of <code>\glentryname</code> 154	<code>\SetDescriptionFootnoteAcronymDisplayStyle:</code>
<code>\glsseeitemformat:</code> new 154	expanded options link op-
<code>\glssortnumberfmt:</code> new 10	tions 177
<code>\glsstepentry:</code> new 160	<code>\setentrycounter:</code> added op-
<code>\glsstepsubentry:</code> new 161	tional argument 167
<code>\glssubentrycounterlabel:</code>	<code>\showacronymlists:</code> new 198
new 161	<code>\showglocounter:</code> new 196
<code>\glssubentryitem:</code> new 162	<code>\showglodesc:</code> new 197

\showglodescplural: new ...	197	\ACRlongpl: made robust	130
\showglofirst: new	195	\Acrlongpl: made robust	129
\showglofirstpl: new	195	\acrlongpl: made robust	129
\showgloflag: new	198	\ACRshort: made robust	124
\showgloindex: new	198	\Acrshort: made robust	123
\showglolevel: new	195	\acrshort: made robust	123
\showglongame: new	197	\ACRshortpl: made robust	126
\showgloparent: new	195	\Acrshortpl: made robust	125
\showgloplural: new	195	\acrshortpl: made robust	125
\showglosort: new	197	\Gls: made robust	93
\showglossaries: new	198	\glsadd: made robust	137
\showglossarycounter: new .	199	\glsaddall: made robust	138
\showglossaryentries: new .	199	\GLSdesc: made robust	109
\showglossaryin: new	198	\Glsdesc: made robust	109
\showglossaryout: new	199	\glsdesc: made robust	108
\showglossarytitle: new ...	199	\GLSdescplural: made robust .	111
\showglosymbol: new	197	\Glsdescplural: made robust .	110
\showglosymbolplural: new .	197	\glsdescplural: made robust .	110
\showglotext: new	195	\glsfirst: made robust	103
\showglotype: new	195	\GLSfirstplural: made robust	106
\showglouseri: new	196	\Glsfirstplural: made robust	106
\showglouserii: new	196	\glsfirstplural: made robust	105
\showglouseriii: new	196	\glslink: made robust	80
\showglouseriv: new	196	\GLSname: made robust	108
\showglouseriv: new	196	\Glsname: made robust	107
\showglouseriv: new	196	\glsname: made robust	107
\showglouseriv: new	197	\GLSpl: made robust	99
subentrycounter: new	9	\Glspl: made robust	97
\writeist: added xindy-only		\glspl: made robust	96
macro definitions to glossary		\GLSplural: made robust	105
open tag	142	\GLSsymbol: made robust	112
modified to support new for-		\Glsymbol: made robust	112
mat	140	\glsymbol: made robust	111
3.01		\GLSsymbolplural: made robust	
\@glswritefiles: added check		114
for empty glossaries	148	\Glsymbolplural: made robust	
General: made robust	95	113
\ACRfull: made robust	173	\glsymbolplural: made robust	
\Acrfull: made robust	173	113
\acrfull: made robust	172	\Glstext: made robust	102
\acrfullformat: removed		\glstext: made robust	101
\acronymfont as it should al-		\GLSuseri: made robust	115
ready be set in the second ar-		\Glsuseri: made robust	115
gument.	173	\glsuseri: made robust	114
\ACRfullpl: made robust	175	\GLSuserii: made robust	117
\Acrfullpl: made robust	174	\Glsuserii: made robust	116
\acrfullpl: made robust	174	\glsuserii: made robust	116
\ACRlong: made robust	128	\GLSuseriii: made robust	118
\Acrlong: made robust	127	\Glsuseriii: made robust	117
\acrlong: made robust	127		

\glsuseriii: made robust	117	\glsresetsubentrycounter:	
\GLSuseriv: made robust	119	new	160
\Glsuseriv: made robust	119	\ifglshaschildren: new	45
\glsuseriv: made robust	118	\ifglshasparent: new	45
\GLSuseriv: made robust	121	\makeglossaries: added list	
\Glsuseriv: made robust	120	parser	147
\glsuseriv: made robust	120	indexonlyfirst: new	19
\GLSuserivi: made robust	122	\printglossary: add a way to	
\Glsuserivi: made robust	122	fetch current entry label . . .	156
\glsuserivi: made robust	121	\renewglossarystyle: new . .	168
		\showglossaryentries: fixed	
3.02		misspelt command	199
\@@do@wrglossary: changed		\SmallNewAcronymDef: fixed	
\@glslocref to \theglsentrycounter		broken short and long plural	187
.	152		
\@do@wrglossary: changed		3.03	
\@do@wr@glossary to test for		\@gls@sanitizesort: new	17
indexonlyfirst option; put old		\@gls@setupsort@standard:	
\@do@wr@glossary code into		used \@gls@sanitizesort . .	10
\@@do@wrglossary	150	General: allow title to set to title	158
\@gls@missingnumberlist:		\glsinlinedescformat: new . .	213
new	54	\glsinlineemptydescformat:	
\@gls@writefiles: added check		new	213
for existence of token in case		\glsinlinenameformat: new . .	213
\makeglossaries has been		\glsinlinepostchild: new . .	213
omitted	148	\glsinlinesubdescformat:	
\@wrglossary: added check for		new	213
glossary file defined	149	\glsinlinesubnameformat:	
General: added check for polyglos-		new	213
sia	23	\glspostinline: replaced “.”	
reversed order of package check	27	with \glspostdescription	213
savenumberlist: new	7	altlongragged4col: added	
ucmark: new	9	check for glsnogroupskip . .	227
\gls@defglossaryentry: added		altsuperragged4col: added	
numberlist element	66	check for glsnogroupskip . .	243
\gls@save@numberlist: new . .	154	alttree: added check for	
\glsdisplaynumberlist: new	136	glsnogroupskip	251
\glsentrycounter: set default		index: added check for	
value	82	glsnogroupskip	245
\Glsentryfull: fixed bug (re-		nogroupskip: new	8
placed \glsentryshortpl		long: added check for	
with \glsentryshort)	135	glsnogroupskip	217
\glsentryfullpl: fixed bug (re-		long3col: added check for	
placed \glsentryshort with		glsnogroupskip	219
\glsentryshortpl)	135	long4col: added check for	
\glsentrynumberlist: new . .	136	glsnogroupskip	220
\glsmoveentry: new	67	longragged: added check for	
\glsnumlistlastsep: new . . .	137	glsnogroupskip	224
\glsnumlistsep: new	137	longragged3col: added check	
		for glsnogroupskip	225

nopostdot:new	8	\PrintChanges:new	5
\printglossary: allow title to override default toctitle	155	\printglossary: Moved aux write to end of document to prevent unwanted whatsit oc- curring here.	156
tree: added check for glsnogroupskip	247		
treenoname: added check for glsnogroupskip	248	3.05	
super: added check for glsnogroupskip	233	\@@do@wrglossary: add Roman case. Fixed bugs in the else statements	151
super3col: added check for glsnogroupskip	235	\@gls@link: added check for “no- hypertypes”	81
super4col: added check for glsnogroupskip	236	\@gls@nohyperlist:new	15
superragged: added check for glsnogroupskip	240	mcolalttree: replaced ‘2’ with \glsmcols	231
superragged3col: added check for glsnogroupskip	241	mcolindex: replaced ‘2’ with \glsmcols	229
3.04		mcoltree: replaced ‘2’ with \glsmcols	229
\@@do@wrglossary: changed \theglsentrycounter back to \@glslocref	152	mcoltreenoname: replaced ‘2’ with \glsmcols	230
modified to compensate for possible incorrect page num- ber	150	\gls@protected@pagefmts: added Roman to list	150
\@gls@escbsdq: unsani- tize \gls@numberpage, \gls@alphpage,\gls@Alphpage and \gls@romanpage	84	\gls@Romanpage:new	150
General: Added check for doc package	4	\GlsDeclareNoHyperList:new	15
added datatool-base as a re- quired package	4	\glsgetgrouplabel: fixed bug (typo in \equal)	166
added local key	80	\nopostdesc: made robust	27
\gls@Alphpage:new	150	nohypertypes:new	15
\gls@alphpage:new	150	3.06	
\gls@disablepagerefexpansion: new	150	\@xdy@main@language: Changed back to using \languagename	21
\gls@numberpage:new	150	\findrootlanguage: Obsolete	41
\gls@protected@pagefmts: new	150	3.07	
\gls@romanpage:new	150	\@gls@link: fixed bug that failed to find entry in list	81
\glsdefmain: added check for doc package	12	\glossarypreamble: modified to work with \setglossarypreamble	30
\glsorg@endtheglossary:new	5	\gls@docclearpage: added check for openright	33
\glsorg@glossary:new	4	\glspostdescription: Added spacefactor code	8
\glsorg@theglossary:new	5	\GlsSetXdyCodePage: Added check for fontspec	42
\glsorg@wrglossary:new	4	\SetDescriptionAcronymDisplayStyle: now using \glsdoparenifnotempty	181
altlist: replaced \newline with paragraph break	215	\setglossarypreamble:new ..	31

3.08a

`\@glo@storeentry`: no longer need to check for special characters in any of the fields other than sort 68
 updated for `\glossentry` 68
`\@glossaryentryfield:switched` to `\glossentry` 67
`\@glossarysubentryfield:` switched to `\subglossentry` 67
 General: added `nogroupskip` key to `\printglossary` 159
 removed definition of `\@glossaryentryfield` .. 287
 removed definition of `\@glossarysubentryfield` 287
`\compatibleglossentry`: new 162
`\compatiblesubglossentry:` new 164
`\glossaryentryfield:` deprecated 164
`\Glossentrydesc`: new 163
`\glossentrydesc`: new 163
`\Glossentryname`: new 163
`\glossentryname`: new 163
`\Glossentrysymbol`: new 164
`\glossentrysymbol`: new 163
`\gls@assign@desc@field`: new 16
`\gls@assign@descplural@field:` new 16
`\gls@assign@field`: new 56
`\gls@ifnotmeasuring`: new ... 69
`\glsaddallunused`: new 138
`\glsexpandfields`: new 56
`\glsnoexpandfields`: new 56
`\glssee`: made robust 153
`\glsseeformat`: made robust .. 153
`\glsseeitem`: made robust 154
`\glsseelist`: made robust 153
`\ifglsdescsuppressed`: new .. 45
`\ifglshasdesc`: new 45
`\ifglshassymbol`: new 46
`list`: updated list style to use `\glossentry` and `\subglossentry` 214
`listdotted`: updated listdotted style to use `\glossentry` and `\subglossentry` 216

`altlist`: updated altlist style to use `\glossentry` and `\subglossentry` 215
`altlongragged4col`: updated to use `\glossentry` and `\subglossentry` 226
`alttree`: updated to use `\glossentry` and `\subglossentry` 249
`index`: added paragraph break at end of environment 245
 updated to use `\glossentry` and `\subglossentry` 245
`inline`: updated inline style to use `\glossentry` and `\subglossentry` 211
`long`: updated to use `\glossentry` and `\subglossentry` 217
`longragged`: updated to use `\glossentry` and `\subglossentry` 223
`longragged3col`: updated to use `\glossentry` and `\subglossentry` 225
`tree`: updated to use `\glossentry` and `\subglossentry` 246
`\setglossarystyle`: new 167
`\setglossentrycompatibility:` new 164
`superragged`: updated to use `\glossentry` and `\subglossentry` 239

3.09a

`\@gls@assign@symbolplural@field:` new 17
`\@gls@default@value`: new ... 51
`\Glsentrydesc`: made robust .. 131
`\Glsentrydescplural`: made robust 132
`\Glsentryfirst`: made robust . 133
`\Glsentryfirstplural`: made robust 133
`\Glsentryfull`: made robust .. 135
`\Glsentryfullpl`: made robust 136
`\Glsentrylong`: made robust .. 135
`\Glsentrylongpl`: made robust 135
`\Glsentryname`: made robust .. 131
`\Glsentryplural`: made robust 132
`\Glsentryshort`: made robust . 135

\Glsentryshortpl: made robust		\@Acrlong: added \glslabel,
.....	135	\glsifplural, \glscapscase,
\Glsentrysymbol: made robust	132	\glsinsert and \glscustomtext
\Glsentrysymbolplural: made	
robust	133	287
\Glsentrytext: made robust ..	132	\@Acrshort: added \glslabel,
\Glsentryuseri: made robust .	133	\glsifplural, \glscapscase,
\Glsentryuserii: made robust	134	\glsinsert and \glscustomtext
\Glsentryuseriii: made robust	
.....	134	285
\Glsentryuseriv: made robust	134	\@GLS@: add \glslabel,
\Glsentryuserv: made robust .	134	\glsifplural, \glscapscase,
\Glsentryuservi: made robust	134	\glscustomtext and
\glstextup: new	172	\glsinsert
\if@gls@docloaded: Add a fix for		95
\RecordChanges	4	change to using \glsentryfmt
\ifglshassymbol: changed test		style commands
to check for \@gls@default@symbol		95
.....	46	removed \MakeUppercase
3.10a		(now moved to \glsentryfmt)
\@gls@keymap: new	58
\@gls@provide@newglossary:		95
new	47	\@GLSpl: add \glslabel,
\@glsdefaultplural: Obsolete .	54	\glsifplural, \glscapscase,
\@glsnodelsc: new	53	\glscustomtext and
\gls@assign@type@field: new	16	\glsinsert
\gls@defglossaryentry:		99
Changed to using \@gls@default@value		change to using \glsentryfmt
.....	62	style commands
new	62	99
\gls writedefhook: new	61	removed \MakeUppercase
\makeglossaries: Added		as now dealt with in
providecommand code to aux		\glsentryfmt
file	146	99
\new@glossaryentry: new	56	\@Gls@: add \glsifplural,
\newglossary: added \@gls@provide@newglossary		\glscapscase, \glscustomtext
.....	48	and \glsinsert
\printglossary: Added provide-		94
command code to aux file ..	157	change to using \glsentryfmt
3.11a		style commands
\@ACRlong: added \glslabel,		94
\glsifplural, \glscapscase,		removed \makefirstuc (now
\glsinsert and \glscustomtext		dealt with in \glsentryfmt)
.....	287	94
\@ACRshort: added \glslabel,		\@GLSpl@: add \glsifplural,
\glsifplural, \glscapscase,		\glscapscase, \glscustomtext
\glsinsert and \glscustomtext		and \glsinsert
.....	286	98
		change to using \glsentryfmt
		style commands
		98
		removed \makefirstuc (now
		dealt with in \glsentryfmt)
		98
		\@acrlong: added \glslabel,
		\glsifplural, \glscapscase,
		\glsinsert and \glscustomtext
	
		286
		\@acrshort: added \glslabel,
		\glsifplural, \glscapscase,
		\glsinsert and \glscustomtext
	
		285
		\@gls@: add \glslabel,

<code>\glsifplural, \glscapscase,</code>	changed to just use <code>\glsentrysymbolplural</code>
<code>\glscustomtext</code> and 113, 114
<code>\glsinsert</code> 93	changed to just use <code>\Glsentrysymbol</code>
change to using <code>\glsentryfmt</code> 112
style commands 93	changed to just use <code>\glsentrysymbol</code>
<code>\@gls@noexpand@fields</code> : Fixed 112, 113
bug expand replaced with	Changed to just use
<code>noexpand</code> 55	<code>\Glsentrytext</code> 102
<code>\@glsdisp</code> : add <code>\glslabel,</code>	changed to just use <code>\glsentrytext</code>
<code>\glsifplural, \glscapscase,</code> 102
<code>\glscustomtext</code> and	changed to just use <code>\Glsentryuseriii</code>
<code>\glsinsert</code> 100 118
change to using <code>\glsentryfmt</code>	changed to just use <code>\glsentryuseriii</code>
style commands 101 117, 118
<code>\@glspl@</code> : add <code>\glslabel,</code>	changed to just use <code>\Glsentryuserii</code>
<code>\glsifplural, \glscapscase,</code> 116
<code>\glscustomtext</code> and	changed to just use <code>\glsentryuserii</code>
<code>\glsinsert</code> 96 116, 117
General: added <code>\glslabel,</code>	changed to just use <code>\Glsentryuseriv</code>
<code>\glsifplural, \glscapscase,</code> 119
<code>\glsinsert</code> and <code>\glscustomtext</code>	changed to just use <code>\glsentryuseriv</code>
..... 123–131 120
changed to just use <code>\Glsentrydescplural</code>	changed to just use <code>\Glsentryuseri</code>
..... 111 115
changed to just use <code>\glsentrydescplural</code>	changed to just use <code>\glsentryuseri</code>
..... 110, 111 115
changed to just use <code>\Glsentrydesc</code>	changed to just use <code>\Glsentryuservi</code>
..... 109 122
changed to just use <code>\glsentrydesc</code>	changed to just use <code>\glsentryuservi</code>
..... 109, 110 122, 123
changed to just use <code>\Glsentryfirstplural</code>	changed to just use <code>\Glsentryuserv</code>
..... 106 121
changed to just use <code>\glsentryfirstplural</code>	changed to just use <code>\glsentryuserv</code>
..... 106, 107 120, 121
changed to just use <code>\Glsentryfirst</code>	Now requires textcase 3
..... 103	acronymlists: replaced
changed to just use <code>\glsentryfirst</code>	<code>\@addtoacronymlists</code> with
..... 103, 104	<code>\DeclareAcronymList</code> 14
changed to just use <code>\Glsentryname</code>	<code>\defglsdisplay</code> : obsoleted 78
..... 108	<code>\defglsdisplayfirst</code> : obso-
changed to just use <code>\glsentryname</code>	leted 79
..... 107, 108	<code>\defglsentryfmt</code> : new 47
changed to just use <code>\Glsentryplural</code>	<code>\forGlsentries</code> : replaced <code>\ifx</code>
..... 105	with <code>\ifdefempty</code> 43
changed to just use <code>\glsentryplural</code>	<code>\gls@assign@desc</code> : new 61
..... 104, 105	<code>\gls@defglossaryentry</code> : Fixed
changed to just use <code>\Glsentrysymbolplural</code>	default counter if none sup-
..... 114	plied 65
	<code>\gls@doentryfmt</code> : new 47

<code>\glstentryfull</code> : changed to use	<code>super3colheader</code> : switched to
<code>\acrfullformat</code> 135	<code>\tabularnewline</code> 235
<code>\Glstentryfullpl</code> : changed to	<code>super4col</code> : switched to
use <code>\acrfullformat</code> 136	<code>\tabularnewline</code> 236
<code>\glstentryfullpl</code> : changed to	<code>super4colheader</code> : switched to
use <code>\acrfullformat</code> 135	<code>\tabularnewline</code> 236
<code>\glsglossarymark</code> : renamed	<code>super4colheaderborder</code> :
<code>\glossarymark</code> to <code>\glsglossarymark</code>	switched to <code>\tabularnewline</code>
to avoid conflict with memoir 31 237
<code>\glsprestandardssort</code> : new 9	<code>superheader</code> : switched to
<code>\glsssetnoexpandfield</code> : new .. 16	<code>\tabularnewline</code> 233
<code>altsuper4colheader</code> : switched	<code>superheaderborder</code> : switched to
to <code>\tabularnewline</code> 238	<code>\tabularnewline</code> 234
<code>altsuper4colheaderborder</code> :	
switched to <code>\tabularnewline</code>	3.14a
..... 238	<code>\@glswritefiles</code> : renamed
<code>long</code> : switched to <code>\tabularnewline</code>	<code>\glswritefiles</code> to <code>\@glswritefiles</code>
..... 217	and used “savewrites” option
<code>long3col</code> : switched to <code>\tabularnewline</code>	to set <code>\glswritefiles</code> 148
..... 219	General: new 200
<code>long3colheader</code> : switched to	acronyms: new 13
<code>\tabularnewline</code> 219	<code>\gls@defglossaryentry</code> : added
<code>long3colheaderborder</code> : switched	check for existence of default
to <code>\tabularnewline</code> 220	glossary 63
<code>long4col</code> : switched to <code>\tabularnewline</code>	set the default for firstplural to
..... 220	be the value of plural 65
<code>long4colheader</code> : switched to	<code>xindygloss</code> : new 21
<code>\tabularnewline</code> 221	<code>\longprovideglossaryentry</code> :
<code>longheader</code> : switched to	new 62
<code>\tabularnewline</code> 218	<code>compatible-2.07</code> : added check
<code>longheaderborder</code> : switched to	for 2.07 before setting 3.07
<code>\tabularnewline</code> 218	compatibility 22
<code>\SetFootnoteAcronymDisplayStyle</code> :	<code>notranslate</code> : new 19
fixed missing argument bug 184	<code>\provideglossaryentry</code> : new . 56
<code>super</code> : switched to <code>\tabularnewline</code>	4.0
..... 233	<code>\gls@defglossaryentry</code> : added
<code>super3col</code> : switched to	check for first key 65
<code>\tabularnewline</code> 234	

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols	
\@do@wrglossary	150
\@glossarysec	5
\@glossaryseclabel	6
\@glossarysecstar	6
\@gls@default@entryfmt	281
\@gls@expand@field	55
\@ACRlong	287
\@ACRshort	286
\@Acrlong	286
\@Acrshort	285
\@GLS@	95
\@GLSpl	99
\@Gls@	94
\@Glspl@	98
\@PGLS	205
\@PGLS@	205
\@PGLSpl	206
\@PGLSpl@	206
\@PglS	204
\@PglS@	204
\@PglSpl	204
\@PglSpl@	205
\@acrlong	286
\@acrshort	285
\@addtoacronymlists	13
\@delimN	169
\@delimR	169
\@disable@onlypremakeg	24
\@disable@premakecs	25
\@disabled@glSaddxdycounters	35
\@do@seeglossary	152
\@do@wrglossary	150, 253
\@glo@seeautonumberlist	7
\@glo@storeentry	68
\@glo@types	47
\@glossary	149
\@glossary@default@style	6
\@glossaryentryfield	67
\@glossarysection	32
\@glossarysubentryfield	67
\@gls	92
\@gls@	92
\@gls@@link	81
\@gls@addpredefinedattributes	37
\@gls@assign@symbol@field ...	17
\@gls@assign@symbolplural@field	17
\@gls@checkactual	89
\@gls@checkboxar	88
\@gls@checkescactual	86
\@gls@checkescbar	87
\@gls@checkesclevel	87
\@gls@checkescquote	86
\@gls@checklevel	88
\@gls@checkmkidxchars	85
\@gls@checkquote	85
\@gls@codepage	42
\@gls@counterwithin	9
\@gls@default@value	51
\@gls@do@acronymsdef	13
\@gls@escbsdq	84
\@gls@expand@fields	55
\@gls@fixbraces	153
\@gls@getcounter	49
\@gls@getcounterprefix	152
\@gls@getgrouptitle	166
\@gls@hypergroup	210
\@gls@ifinlist	35
\@gls@keymap	58, 200
\@gls@link	81
\@gls@loadlist	8
\@gls@loadlong	7
\@gls@loadsuper	7
\@gls@loadtree	8
\@gls@makefirsttuc	208
\@gls@missingnumberlist	54
\@gls@noaccess	275
\@gls@noexpand@field	54
\@gls@noexpand@fields	54
\@gls@nohyperlist	15
\@gls@notranslatorhook	18
\@gls@onlypremakeg	24
\@gls@provide@newglossary ...	47
\@gls@renewglossary	149

<code>\defglstdisplayfirst</code>	79
<code>\defglstentry</code>	48
<code>\defglstentryfmt</code>	47, 50, 51, 73
<code>\DefineAcronymSynonyms</code>	192
<code>\delimN</code>	30, 168
<code>\delimR</code>	30, 168
<code>description</code> (environment)	213, 214
<code>description</code> (key)	50
<code>description</code> (option)	20
<code>descriptionaccess</code> (key)	274
<code>\DescriptionDUANewAcronymDef</code>	180
<code>\DescriptionFootnoteNewAcronymDef</code>	178, 289
<code>\descriptionname</code>	25
<code>\DescriptionNewAcronymDef</code>	182, 290
<code>descriptionplural</code> (key)	50
<code>descriptionpluralaccess</code> (key)	274
<code>doc</code> package	4, 5, 12
<code>dua</code> (option)	20
<code>\DUANewAcronymDef</code>	189
E	
<code>entrycounter</code> (option)	9
<code>entrycounterwithin</code> (option)	9
<code>\entryname</code>	25
environments:	
<code>align</code>	69, 82
<code>description</code>	213, 214
<code>longtable</code>	7, 194, 216–228
<code>multicols</code>	228
<code>supertabular</code> ...	8, 194, 232–244
<code>theglossary</code>	5, 30, 31, 162, 167, 230, 231, 246, 247, 249
<code>theindex</code>	245
<code>equation counter</code>	82, 83
<code>etoolbox</code> package	4, 206
F	
file types	
<code>.aux</code>	156
<code>.glo</code>	68
<code>.ist</code>	138, 139, 145
<code>.toc</code>	34
<code>.xdy</code>	28
<code>glo</code>	199
<code>\findrootlanguage</code>	41
<code>first</code> (key)	50
<code>firstaccess</code> (key)	273
<code>\firstacronymfont</code>	175

<code>firstplural</code> (key)	51
<code>firstpluralaccess</code> (key)	273
<code>footnote</code> (option)	19
<code>\FootnoteNewAcronymDef</code> ..	184, 291
<code>\forallallglossaries</code>	43
<code>\forallallglstentries</code>	43
<code>\forallrglstentries</code>	43
G	
<code>garamondx</code> package	172
<code>\glolinkprefix</code>	82
<code>glossareentry counter</code>	160
<code>glossaries</code> package	
... ..	42, 139, 194, 200, 213, 252, 272
<code>glossaries-accsupp</code> package ...	67, 272
<code>\GlossariesWarning</code>	15
<code>\GlossariesWarningNoLine</code>	15
<code>\glossary</code>	47, 145, 149, 166
<code>glossary counters:</code>	
<code>glossaryentry</code>	160
<code>glossarysubentry</code>	160
<code>glossary keys:</code>	
<code>access</code>	273
<code>counter</code>	51
<code>description</code>	50
<code>descriptionaccess</code>	274
<code>descriptionplural</code>	50
<code>descriptionpluralaccess</code> ..	274
<code>first</code>	50
<code>firstaccess</code>	273
<code>firstplural</code>	51
<code>firstpluralaccess</code>	273
<code>long</code>	53
<code>longaccess</code>	274
<code>longplural</code>	53
<code>longpluralaccess</code>	274
<code>name</code>	50
<code>nonumberlist</code>	52
<code>parent</code>	52
<code>plural</code>	50
<code>pluralaccess</code>	273
<code>see</code>	52
<code>short</code>	53
<code>shortaccess</code>	274
<code>shortplural</code>	53
<code>shortpluralaccess</code>	274
<code>sort</code>	50
<code>symbol</code>	51
<code>symbolaccess</code>	274

symbolplural	51	altsuperragged4col	242–244, 270
symbolpluralaccess	274	altsuperragged4col	242
text	50	altsuperragged4colborder	
textaccess	273	244, 270
type	51	altsuperragged4colborder	244
user1	52	altsuperragged4colheader	
user2	52	243, 270
user3	52	altsuperragged4colheader	243
user4	52	altsuperragged4colheaderborder	
user5	53	244, 270
user6	53	altsuperragged4colheaderborder	
glossary package	1, 171	244
glossary styles:		alttree	231, 249, 251, 266
altlist	215, 260	alttree	249
altlist	215	alttreegroup	252, 267
altlistgroup	215, 260	alttreegroup	251
altlistgroup	215	alttreehypergroup ...	252, 267
altlisthypergroup ...	215, 260	alttreehypergroup	252
altlisthypergroup	215	index	228, 244–246, 264
altlong4col ..	221, 222, 226, 262	index	244
altlong4col	221	indexgroup	245, 246, 265
altlong4colborder ...	222, 262	indexgroup	245
altlong4colborder	222	indexhypergroup	246, 265
altlong4colheader ...	222, 262	indexhypergroup	246
altlong4colheader	222	inline	259
altlong4colheaderborder .		inline	211
.....	222, 263	list	6, 213–216, 260
altlong4colheaderborder .	222	list	213
altlongragged4col	226, 227, 264	listdotted	216, 260
altlongragged4col	226	listdotted	216
altlongragged4colborder .		listgroup	214, 260
.....	227, 264	listgroup	214
altlongragged4colborder .	227	listhypergroup	214, 260
altlongragged4colheader .		listhypergroup	214
.....	227, 264	long	217, 218, 223, 261, 263
altlongragged4colheader .	227	long	217
altlongragged4colheaderborder		long3col	218, 219, 261
.....	228, 264	long3col	218
altlongragged4colheaderborder		long3colborder	219, 261
.....	228	long3colborder	219
altsuper4col ..	237, 238, 242, 271	long3colheader	219, 262
altsuper4col	237	long3colheader	219
altsuper4colborder ..	238, 272	long3colheaderborder	219, 262
altsuper4colborder	238	long3colheaderborder ...	219
altsuper4colheader ..	238, 272	long4col	220, 221, 262
altsuper4colheader	238	long4col	220
altsuper4colheaderborder		long4colborder	221, 262
.....	238, 272	long4colborder	221
altsuper4colheaderborder	238	long4colheader	220, 262

long4colheader	220	mcoltreenonamehypergroup	
long4colheaderborder	221, 262	230, 268
long4colheaderborder	mcoltreenonamehypergroup	230
longborder	217, 261	sublistdotted	261
longborder	217	sublistdotted	216
longheader	218, 261	super	232–234, 240, 270
longheader	218	super	232
longheaderborder	218, 261	super3col	234, 235, 270
longheaderborder	218	super3col	234
longragged	223–225	super3colborder	235, 271
longragged	223	super3colborder	235
longragged3col ...	225, 226, 263	super3colheader	235, 271
longragged3col	225	super3colheader	235
longragged3colborder	225, 263	super3colheaderborder	235, 271
longragged3colborder	225	super3colheaderborder ...	235
longragged3colheader	226, 263	super4col	236, 237, 271
longragged3colheader	226	super4col	236
longragged3colheaderborder		super4colborder	237, 271
.....	226, 264	super4colborder	237
longragged3colheaderborder		super4colheader	236, 271
.....	226	super4colheader	236
longraggedborder	224, 263	super4colheaderborder	237, 271
longraggedborder	224	super4colheaderborder ...	237
longraggedheader	224, 263	superborder	233, 270
longraggedheader	224	superborder	233
longraggedheaderborder	224, 263	superheader	233, 270
longraggedheaderborder ..	224	superheader	233
mcolalttree	231, 268	superheaderborder ...	234, 270
mcolalttree	231	superheaderborder	234
mcolalttreegroup	231, 268	superragged	239–241, 269
mcolalttreegroup	231	superragged	239
mcolalttreehypergroup	231, 268	superragged3col ..	241, 242, 269
mcolalttreehypergroup ...	231	superragged3col	241
mcolindex	229, 267	superragged3colborder	241, 269
mcolindex	228	superragged3colborder ...	241
mcolindexgroup	229, 268	superragged3colheader	242, 269
mcolindexgroup	229	superragged3colheader ...	242
mcolindexhypergroup .	229, 268	superragged3colheaderborder	
mcolindexhypergroup	229	242, 269
mcoltree	229, 268	superraggedborder ...	240, 269
mcoltree	229	superraggedborder	240
mcoltreegroup	268	superraggedheader ...	240, 269
mcoltreegroup	229	superraggedheader	240
mcoltreehypergroup ..	230, 268	superraggedheaderborder .	
mcoltreehypergroup	230	240, 269
mcoltreenoname	230, 268	superraggedheaderborder .	240
mcoltreenoname	230	superraggedright3colheaderborder	
mcoltreenonamegroup .	230, 268	242
mcoltreenonamegroup	230	tree	229, 246–249, 265

tree	246	105, 107, 108, 110, 111, 113,	
treegroup	230, 247, 265	114, 116–118, 120, 121, 161, 202	
treegroup	247	\gls@Alphpage	150
treehypergroup	247, 265	\gls@alphpage	150
treehypergroup	247	\gls@assign@desc	61
treenoname	230, 248, 266	\gls@assign@desc@field	16
treenoname	248	\gls@assign@descplural@field	16
treenonamegroup	249, 266	\gls@assign@field	56
treenonamegroup	248	\gls@assign@name@field	16
treenonamehypergroup	249, 266	\gls@assign@type@field	16
treenonamehypergroup	249	\gls@checkisacronymlist	14
glossary-hypernav package	139	\gls@checkseeallowed	52
glossary-list package	6, 8, 213	\gls@codepage	21
glossary-long package		\gls@defglossaryentry	62
.....	7, 216, 217, 226, 232	\gls@disablepagerefexpansion	150
glossary-longragged package	223	\gls@doclearpage	33
glossary-mcols package	228	\gls@doentryfmt	47
glossary-super package		\gls@hypergroup prerun	210
.....	7, 8, 217, 232, 239, 242	\gls@ifnotmeasuring	69
glossary-superragged package	239	\gls@level	54
glossary-tree package	8, 244	\gls@numberpage	150
glossaryentry (counter)	160	\gls@protected@pagefmts	150
glossaryentry counter	9, 160, 161	\gls@Romanpage	150
\glossaryentryfield	164, 168	\gls@romanpage	150
\glossaryentrynumber	159	\gls@save@numberlist	154
\glossaryentrynumbers		\gls@suffixF	29
.....	7, 30, 155, 157	\gls@suffixFF	29
\glossaryheader	162, 167	\glsaccessdisplay	281
\glossarymark	32	\glsaccsupp	278
\glossaryname	25, 26	\glsadd	71, 137, 166
\glossarypostamble	31, 168	\glsadd options	
\glossarypreamble	30, 168	counter	137
\glossarysection	6, 31, 47	format	137, 168
\glossarystyle	167, 194	\glsaddall	71, 138
glossarysubentry (counter) ...	160	\glsaddall options	
glossarysubentry counter ...	9, 160–162	types	137, 138
\glossarysubentryfield	164	\glsaddallunused	138
\glossentry	51, 162	\glsaddkey	58
\Glossentrydesc	163	\GlsAddLetterGroup	43
\glossentrydesc	163, 287	\GlsAddSortRule	41
\Glossentryname	163	\GlsAddXdyAlphabet	37
\glossentryname	163, 287	\GlsAddXdyAttribute	36, 252
\Glossentrysymbol	164	\GlsAddXdyCounters	35, 253
\glossentrysymbol	163, 287	\GlsAddXdyLocation	39, 253
\GLS	95	\GlsAddXdyStyle	41
\Gls	93, 97, 207	\glsautoprefix	6
\gls	4,	\glsclearpage	34
	51, 71, 80, 92, 95, 96, 101, 103–	\glsclosebrace	139
		\glscompositor	29, 39

<code>\glscounter</code>	15, 48	<code>\glstentrylongpluralaccess</code> ..	278
<code>\GlsDeclareNoHyperList</code>	15	<code>\Glsentryname</code>	131
<code>\glsdefaulttype</code>	12	<code>\glstentryname</code>	131, 154
<code>\glsdefmain</code>	12	<code>\glstentrynumberlist</code>	136
<code>\GLSdesc</code>	109	<code>\Glsentryplural</code>	132
<code>\Glsdesc</code>	109	<code>\glstentryplural</code>	132
<code>\glsdesc</code>	108, 109	<code>\glstentrypluralaccess</code>	277
<code>\GLSdescplural</code>	111	<code>\Glsentryprefix</code>	201
<code>\Glsdescplural</code>	110	<code>\glstentryprefix</code>	201
<code>\glsdescplural</code>	110, 111	<code>\Glsentryprefixfirst</code>	201
<code>\glsdescriptionaccessdisplay</code>	279	<code>\glstentryprefixfirst</code>	201
<code>\glsdescriptionpluralaccessdisplay</code>	280	<code>\Glsentryprefixfirstplural</code> .	201
.....	280	<code>\glstentryprefixfirstplural</code> .	201
<code>\glsdescwidth</code> ...	217, 223, 232, 239	<code>\Glsentryprefixplural</code>	202
<code>\glsdisablehyper</code>	91	<code>\glstentryprefixplural</code>	201
<code>\glsdisp</code>	100	<code>\Glsentryshort</code>	135
<code>\glsdisplay</code>	72, 78, 92	<code>\glstentryshort</code>	134
<code>\glsdisplayfirst</code>	72, 78, 92	<code>\glstentryshortaccess</code>	277
<code>\glsdisplaynumberlist</code>	136	<code>\Glsentryshortpl</code>	135
<code>\glsdoifexists</code>	44	<code>\glstentryshortpl</code>	135
<code>\glsdoifnoexists</code>	45	<code>\glstentryshortpluralaccess</code> .	277
<code>\glsdoparenifnotempty</code>	186	<code>\glstentrysort</code>	133
<code>\glsenablehyper</code>	92	<code>\Glsentrysymbol</code>	132
<code>\glstentryaccess</code>	276	<code>\glstentrysymbol</code>	132
<code>\glstentrycounter</code>	82	<code>\glstentrysymbolaccess</code>	277
<code>\glstentrycounterlabel</code>	161	<code>\Glsentrysymbolplural</code>	133
<code>\Glsentrydesc</code>	131	<code>\glstentrysymbolplural</code>	132
<code>\glstentrydesc</code>	131	<code>\glstentrysymbolpluralaccess</code>	277
<code>\glstentrydescaccess</code>	277	<code>\Glsentrytext</code>	132
<code>\Glsentrydescplural</code>	132	<code>\glstentrytext</code>	132, 154
<code>\glstentrydescplural</code>	132	<code>\glstentrytextaccess</code>	276
<code>\glstentrydescpluralaccess</code> ..	277	<code>\glstentrytype</code>	133
<code>\Glsentryfirst</code>	133	<code>\Glsentryuseri</code>	133
<code>\glstentryfirst</code>	133	<code>\glstentryuseri</code>	133
<code>\glstentryfirstaccess</code>	276	<code>\Glsentryuserii</code>	134
<code>\Glsentryfirstplural</code>	133	<code>\glstentryuserii</code>	134
<code>\glstentryfirstplural</code>	133	<code>\Glsentryuseriii</code>	134
<code>\glstentryfirstpluralaccess</code> .	277	<code>\glstentryuseriii</code>	134
<code>\glstentryfmt</code>	50, 51, 72	<code>\Glsentryuseriv</code>	134
<code>\Glsentryfull</code>	135	<code>\glstentryuseriv</code>	134
<code>\glstentryfull</code>	135	<code>\Glsentryuserv</code>	134
<code>\Glsentryfullpl</code>	136	<code>\glstentryuserv</code>	134
<code>\glstentryfullpl</code>	135	<code>\Glsentryuservi</code>	134
<code>\glstentryitem</code>	161	<code>\glstentryuservi</code>	134
<code>\Glsentrylong</code>	135	<code>\glsexpandfields</code>	56
<code>\glstentrylong</code>	135	<code>\GLSfirst</code>	103
<code>\glstentrylongaccess</code>	277	<code>\Glsfirst</code>	103
<code>\Glsentrylongpl</code>	135	<code>\glsfirst</code>	103
<code>\glstentrylongpl</code>	135	<code>\glsfirstaccessdisplay</code>	279

<code>\GLSfirstplural</code>	106	<code>\Glsname</code>	107
<code>\Glsfirstplural</code>	106	<code>\glsname</code>	107, 108
<code>\glsfirstplural</code>	105, 106	<code>\glsnameaccessdisplay</code>	278
<code>\glsfirstpluralaccessdisplay</code>	279	<code>\glsnamefont</code>	168
<code>\glsgenentryfmt</code>	76	<code>\glsnavhyperlink</code>	209
<code>\glsgetgrouplabel</code>	166	<code>\glsnavhypertarget</code>	209
<code>\glsgetgrouptitle</code>	139, 166	<code>\glsnavigation</code>	210
<code>\glsglossarymark</code>	9, 31	<code>\glsnextpages</code>	159
<code>\glsgroupheading</code>	165, 168	<code>\glsnoexpandfields</code>	56
<code>\glsgroupskip</code>	165, 168, 213	<code>\glsnonextpages</code>	159
<code>\glshyperlink</code>	137	<code>\glsnoxindywarning</code>	34
<code>\glshypernavsep</code>	211	<code>\glsnumberformat</code>	30
<code>\glshyphnumber</code>	30, 169	<code>\glsnumbersgroupname</code>	26, 139, 165
<code>\glsIfListOfAcronyms</code>	14	<code>\glsnumlistlastsep</code>	137
<code>\glsinlinedescformat</code>	213	<code>\glsnumlistsep</code>	137
<code>\glsinlinedopostchild</code>	211, 212	<code>\glsopenbrace</code>	139
<code>\glsinlineemptydescformat</code>	213	<code>\glsorder</code>	20
<code>\glsinlinenameformat</code>	213	<code>\glsorg@endtheglossary</code>	5
<code>\glsinlineparentchildseparator</code>	213	<code>\glsorg@glossary</code>	4
<code>\glsinlinepostchild</code>	213	<code>\glsorg@theglossary</code>	5
<code>\glsinlineseparator</code>	213	<code>\glsorg@wrglossary</code>	4
<code>\glsinlinesubdescformat</code>	213	<code>\glspagelistwidth</code>	217, 223, 232, 239
<code>\glsinlinesubnameformat</code>	213	<code>\glspar</code>	28
<code>\glsinlinesubseparator</code>	213	<code>\GLSpl</code>	99
<code>\glskeylisttok</code>	175	<code>\Glspl</code>	97, 207
<code>\glslabeltok</code>	175	<code>\glspl</code>	71, 96, 97, 99
<code>\glslink</code>	71, 72, 80, 92, 137, 166, 168	<code>\GLSplural</code>	105
<code>\glslink options</code>		<code>\Glsplural</code>	104
counter	80, 92, 200	<code>\glsplural</code>	104, 105
format	80, 92, 168	<code>\glspluralaccessdisplay</code>	278
hyper	80, 92	<code>\glspluralsuffix</code>	26, 50, 51
local	80	<code>\glspostdescription</code>	8
<code>\glslistdottedwidth</code>	216	<code>\glspostinline</code>	213
<code>\glslocalreset</code>	70	<code>\glsprestandardsort</code>	9
<code>\glslocalresetall</code>	71	<code>\glsquote</code>	139
<code>\glslocalunset</code>	70	<code>\glsrefentry</code>	161
<code>\glslocalunsetall</code>	71	<code>\glsreset</code>	69
<code>\glslongaccessdisplay</code>	280	<code>\glsresetall</code>	70
<code>\glslongaccesskey</code>	292	<code>\glsresetentrylist</code>	159
<code>\glslongkey</code>	172	<code>\glsresetsubentrycounter</code>	160
<code>\glslongpluralaccessdisplay</code>	280	<code>\glssee</code>	153
<code>\glslongpluralaccesskey</code>	292	<code>\glsseeformat</code>	141, 153
<code>\glslongpluralkey</code>	172	<code>\glsseeitem</code>	154
<code>\glslongtok</code>	175	<code>\glsseeitemformat</code>	154
<code>\glsmakefirsttuc</code>	208	<code>\glsseelastsep</code>	154
<code>\glsmcols</code>	228	<code>\glsseelist</code>	153
<code>\glsmoveentry</code>	67	<code>\glsseesep</code>	154
<code>\GLSname</code>	108	<code>\glsSetAlphaCompositor</code>	29
		<code>\glsSetCompositor</code>	28, 29

`\ifglstranslate` 18
`\ifglused` 44, 69
`\ifglxindy` 21
`index (style)` 244
`indexgroup (style)` 245
`indexhypergroup (style)` 246
`indexonlyfirst (option)` 19
`inline (style)` 211
`\inputencodingname` 21
`\istfile` 148
`\istfilename` 28
`\item` 168, 213, 245

L

`link text` 72
`list (style)` 213
`listdotted (style)` 216
`listgroup (style)` 214
`listhypergroup (style)` 214
`\loadglsentries` 12, 72
`long (key)` 53
`long (style)` 217
`long3col (style)` 218
`long3colborder (style)` 219
`long3colheader (style)` 219
`long3colheaderborder (style)` .. 219
`long4col (style)` 220
`long4colborder (style)` 221
`long4colheader (style)` 220
`long4colheaderborder (style)` .. 221
`longaccess (key)` 274
`longborder (style)` 217
`longheader (style)` 218
`longheaderborder (style)` 218
`\longnewglossaryentry` 50, 61
`longplural (key)` 53
`longpluralaccess (key)` 274
`\longprovideglossaryentry` ... 62
`longragged (style)` 223
`longragged3col (style)` 225
`longragged3colborder (style)` .. 225
`longragged3colheader (style)` .. 226
`longragged3colheaderborder (style)` 226
`longraggedborder (style)` 224
`longraggedheader (style)` 224
`longraggedheaderborder (style)` 224
`longtable (environment)`
..... 7, 194, 216–228

`longtable package` 216, 223

M

`\makefirstuc` 206
`makeglossaries` 20, 28, 41, 42, 47, 156
`\makeglossaries`
..... 24, 28, 29, 46, 48, 146, 147
`\makeglossary` 147
`makeindex` 306
`makeindex` 9, 21, 25, 28–30, 47, 49,
50, 69, 83, 86, 138, 139, 141,
144, 145, 151, 152, 165, 253, 254
`delim_n` 30
`delim_r` 30
`page_compositor` 28
`special characters` 85, 138
`mcolalttree (style)` 231
`mcolalttreegroup (style)` 231
`mcolalttreehypergroup (style)` . 231
`mcolindex (style)` 228
`mcolindexgroup (style)` 229
`mcolindexhypergroup (style)` ... 229
`mcoltree (style)` 229
`mcoltreegroup (style)` 229
`mcoltreehypergroup (style)` 230
`mcoltreenoname (style)` 230
`mcoltreenonamegroup (style)` ... 230
`mcoltreenonamehypergroup (style)` 230
`memoir class` 149
`mfistuc package` 1
`\mfistucMakeUppercase` 208
`multicol package` 228
`multicols (environment)` 228

N

`name (key)` 50
`\new@glossaryentry` 56
`\newacronym` .. 20, 53, 71, 72, 171, 172
`\newacronymhook` 176
`\newglossary` 15, 47, 49, 145, 147, 158
`\newglossaryentry` 50, 56, 71, 72, 172
`\newglossaryentry options`
`access` 275, 276
`counter` 51
`description` 20, 49, 50,
53, 56, 62, 108, 131, 172, 185, 274
`descriptionaccess` 277, 279
`descriptionplural` 110, 274
`descriptionpluralaccess` .. 277, 280

first	50, 51, 65, 92, 103, 133, 183, 188, 273
firstaccess	276, 279
firstplural	51, 105, 133, 273
firstpluralaccess	277, 279
format	140
long	135, 274
longaccess	277, 280
longplural	135, 274
longpluralaccess	278, 280
name	49, 50, 53, 56, 62, 107, 131, 154, 273
nonumberlist	52
parent	52, 56
plural	50, 65, 104, 273
pluralaccess	277, 278
prefix	201
prefixfirst	201
prefixfirstplural	201
prefixplural	201
see	7, 52, 147
short	134, 274
shortaccess	277, 280
shortplural	135, 274
shortpluralaccess	277, 280
sort	50, 133, 165
symbol ...	49, 51, 111, 132, 179, 181, 183, 188, 220, 236, 273–275
symbolaccess	277, 279
symbolplural	113, 274
symbolpluralaccess	277, 279
text ..	50, 92, 101, 132, 179, 183, 273
textaccess	276, 278
type	12, 51, 71, 133
user1	114, 133, 274
user2	116, 134
user3	117, 134
user4	118, 134
user5	120, 134
user6	121, 134, 274
\newglossarystyle	167
nogroupskip (option)	8
nohypertypes (option)	15
\noist	145, 199, 258
nolist (option)	8
nolong (option)	7
nonumberlist (key)	52
nonumberlist (option)	7
\nopostdesc	27
nopostdot (option)	8
nostyles (option)	8
nosuper (option)	8
notranslate (option)	19
notree (option)	8
numberedsection (option)	6
numberline (option)	5
numbers (option)	22
O	
\oldacronym	171
order (option)	20
P	
package options:	
acronym	12, 13, 25, 158, 172
true	13
acronym	12
acronymlists	14
acronyms	13
compatible-2.07	22
compatible-3.07	22
counter	15
counter	15
description	183, 184
description	20
dua	181, 183, 184
dua	20
entrycounter	160
true	9
entrycounter	9
entrycounterwithin	9
footnote	93–
95, 97–99, 101, 179, 181, 183, 185	
footnote	19
hyperfirst	
false	93–95, 97–99, 101
hyperfirst	19
indexonlyfirst	314
indexonlyfirst	19
makeindex	142, 200
nogroupskip	8
nohypertypes	15
nolist	194
nolist	8
nolong	194, 217
nolong	7
nomain	12
nonumberlist	7
nonumberlist	7

nopostdot	8	\PGLS	205
nostyles	8	\Pgls	203
nosuper	194	\pgls	202
nosuper	8	\PGLSpl	206
notranslate	19	\Pglspl	204
notree	194	\pglspl	203
notree	8	\phantomsection	31–33
numberedsection	6	plural (key)	50
numberline	5	pluralaccess (key)	273
numberline	5	polyglossia package	23, 26
numbers	22	\printacronyms	13
order	20	\PrintChanges	5
sanitize	17, 49, 131, 132	\printglossaries	
sanitize	18	. 12, 30, 46, 49, 147, 155, 158, 209	
sanitizesort	18	\printglossary	
savenumberlist	7	30, 31, 47, 147, 155, 157, 158, 209	
savewrites	22, 311	\printglossary options	
false	146	nogroupskip	159
true	148	nonumberlist	159
savewrites	22	numberedsection	158
section	6, 32	style	158
section	6	title	158
seeautonumberlist	7	toctitle	158
shotcuts	20	type	12, 154, 158
smallcaps	20	\provideglossaryentry	56
smaller	20		
sort		R	
def	9, 10	\renewglossarystyle	168
standard	9	\roman	38
use	9, 10		
sort	9	S	
style	6, 194	sanitize (option)	18
style	6	sanitizesort (option)	18
subentrycounter	160	savenumberlist (option)	7
subentrycounter	9	savewrites (option)	22
symbols	22	section (option)	6
toc	5	see (key)	52
true	5	seeautonumberlist (option)	7
toc	5	\seename	26
translate	19	\SetAcronymLists	14
false	19	\SetAcronymStyle	13, 190
translate	19	\SetCustomDisplayStyle	191
translator	18	\SetCustomStyle	192
ucmark	9	\SetDefaultAcronymDisplayStyle	
xindy	21, 142, 200	176
xindy	21	\SetDefaultAcronymStyle	177
xindygloss	21	\SetDescriptionAcronymDisplayStyle	
\pagelistname	25	181
parent (key)	52	\SetDescriptionAcronymStyle	183

\SetDescriptionDUAAcronymDisplayStyle	\showgloshortaccess	293
..... 180	\showgloshortpluralaccess	293
\SetDescriptionDUAAcronymStyle	\showglosort	197
..... 181	\showglossaries	198
\SetDescriptionFootnoteAcronymDisplayStyle	\showglossarycounter	199
..... 177	\showglossaryentries	199
\SetDescriptionFootnoteAcronymStyle	\showglossaryin	198
..... 179	\showglossaryout	199
\SetDUADisplayStyle	\showglossarytitle	199
..... 189	\showglosymbol	197
\SetDUASStyle	\showglosymbolaccess	293
..... 190	\showglosymbolplural	197
\setentrycounter	\showglosymbolpluralaccess	293
..... 167	\showglotext	195
\SetFootnoteAcronymDisplayStyle	\showglotextaccess	292
..... 184	\showglotype	195
\SetFootnoteAcronymStyle	\showglouserii	196
..... 185	\showglouseriii	196
\setglossarypreamble	\showglouseriv	196
..... 31	\showglouserv	196
\setglossarysection	\showglouservi	197
..... 32	smallcaps (option)	20
\setglossarystyle	smaller (option)	20
..... 167	\SmallNewAcronymDef	187, 291
\setglossentrycompatibility	sort (key)	50
..... 164	sort (option)	9
\SetSmallAcronymDisplayStyle	style (option)	6
..... 186	subentrycounter (option)	9
\SetSmallAcronymStyle	\subglossentry	162
..... 188	\subitem	245
\setStyleFile	sublistdotted (style)	216
..... 28	\subsubitem	245
\setupglossaries	super (style)	232
..... 23	super3col (style)	234
short (key)	super3colborder (style)	235
..... 53	super3colheader (style)	235
shortaccess (key)	super3colheaderborder (style)	235
..... 274	super4col (style)	236
shortplural (key)	super4colborder (style)	237
..... 53	super4colheader (style)	236
shortpluralaccess (key)	super4colheaderborder (style)	237
..... 274	superborder (style)	233
shotcuts (option)	superheader (style)	233
..... 20	superheaderborder (style)	234
\showacronymlists	superragged (style)	239
..... 198	superragged3col (style)	241
\showglodesc	superragged3colborder (style)	241
..... 197	superragged3colheader (style)	242
\showglodescaccess		
..... 293		
\showglodescplural		
..... 197		
\showglodescpluralaccess		
... 293		
\showglofirst		
..... 195		
\showglofirstaccess		
..... 292		
\showglofirsttpl		
..... 195		
\showglofirsttpluralaccess		
.. 293		
\showgloflag		
..... 198		
\showgloindex		
..... 198		
\showglolevel		
..... 195		
\showglolong		
..... 198		
\showglolongaccess		
..... 293		
\showglolongpluralaccess		
... 293		
\showgloname		
..... 197		
\showglonameaccess		
..... 292		
\showgloparent		
..... 195		
\showgloplural		
..... 195		
\showglopluralaccess		
..... 292		
\showgloshort		
..... 198		

superraggedborder (style) 240
 superraggedheader (style) 240
 superraggedheaderborder (style) 240
 superraggedright3colheaderborder
 (style) 242
 supertabular (environment) ...
 8, 194, 232–244
 supertabular package .. 7, 194, 232, 239
 symbol (key) 51
 symbolaccess (key) 274
 \symbolname 25
 symbolplural (key) 51
 symbolpluralaccess (key) 274
 symbols (option) 22

T

text (key) 50
 textaccess (key) 273
 textcase package 3
 \theequation 152
 theglossary (environment)
 5, 30, 31,
 162, 167, 230, 231, 246, 247, 249
 \theHequation 152
 theindex (environment) 245
 toc (option) 5
 \translate 26
 translate (option) 19
 translator package
 23, 26, 155, 294, 303–306
 tree (style) 246

treegroup (style) 247
 treehypergroup (style) 247
 treenoname (style) 248
 treenonamegroup (style) 248
 treenonamehypergroup (style) .. 249
 type (key) 51

U

ucmark (option) 9
 user1 (key) 52
 user2 (key) 52
 user3 (key) 52
 user4 (key) 52
 user5 (key) 53
 user6 (key) 53

W

\warn@nomakeglossaries 146
 \warn@noprintglossary 155
 \writeist 28, 35, 36, 40, 140, 252, 254

X

\xcapitalisewords 208
 \xglsaccsupp 278
 xindy 306
 xindy 9, 21, 28, 29, 34, 37,
 39, 41–43, 68, 89, 90, 139–141,
 151, 152, 156, 165, 199, 253, 254
 xindy (option) 21
 xindygloss (option) 21
 \xmakefirstuc 208
 xspace package 4, 171